SYBASE®
An SAP® Company

New Features Summary

Adaptive Server® Enterprise 15.7 SP100

DOCUMENT ID: DC01165-01-1570100-02
LAST REVISED: July 2013
Copyright © 2013 by Sybase, Inc. All rights reserved.
This publication pertains to Sybase software and to any subsequent release until otherwise indicated in new editions or technical notes. Information in this document is subject to change without notice. The software described herein is furnished under a license agreement, and it may be used or copied only in accordance with the terms of that agreement.
Upgrades are provided only at regularly scheduled software release dates. No part of this publication may be reproduced, transmitted, or translated in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without the prior written permission of Sybase, Inc.
Sybase trademarks can be viewed at the Sybase trademarks page at *http://www.sybase.com/detail?id=1011207*. Sybase and the marks listed are trademarks of Sybase, Inc. ® indicates registration in the United States of America.
SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world.
Java and all Java-based marks are trademarks or registered trademarks of Oracle and/or its affiliates in the U.S. and other countries.
Unicode and the Unicode Logo are registered trademarks of Unicode, Inc.
IBM and Tivoli are registered trademarks of International Business Machines Corporation in the United States, other countries, or both.
All other company and product names mentioned may be trademarks of the respective companies with which they are associated.
Use, duplication, or disclosure by the government is subject to the restrictions set forth in subparagraph (c)(1)(ii) of DFARS 52.227-7013 for the DOD and as set forth in FAR 52.227-19(a)-(d) for civilian agencies.
Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

Version 15.7 SP100 1
Adaptive Server 15.7 SP100 Feature and Platform Matrix 1
Shrinking Databases 3
Enhancements to Backup and Restore 3
Aggregating Metrics from Syntactically Similar Queries 3
Updates to Precomputed Result Sets 4
Improved Data Load Performance 4
Incremental Reorganization 4
Creating Indexes Without Blocking Access to Data 4
Query Plan and Execution Statistics in HTML 5
Gathering Hash-Based Statistics with create index 5
Query Plan Optimization with Bloom Filters 5
Support for Replication by Column Value 5
Support for Multiple Scanner Threads 5
Sybase Central Adaptive Server Plug-in 6
Running Adaptive Server on VMware vSphere 6
Support for OData 6
Support for NTLM and MIT Kerberos on Windows 64-bit 6
Properties Added to Responses File 6
Automatic Physical Database Rearrangement 6
System Changes for Adaptive Server Version 15.7 SP100 7
Commands 7
Functions 8
System Stored Procedures 9
Configuration Parameters 11
System Tables 11
Utilities 12

Monitoring Table Changes 12
Permission Changes for Commands and Functions 14
Version 15.7 ESD #4 17
15.7 ESD #4.2 17
Improved Rollback Reporting 17
15.7 ESD #4 17
Remote Backup Server Maximum Name Length 17
Support for Query Plan Pinning 17
AF_UNIX Support 17
Changes to sp_addthreshold and Roles 17
System Changes for Adaptive Server Version 15.7 ESD #4 18
Version 15.7 ESD #3 19
sybrestore Utility 19
System Changes for Adaptive Server Version 15.7 ESD #3 19
Commands 19
Functions 19
System Stored Procedures 20
Monitoring Table Changes 20
Version 15.7 ESD #2 21
Granular Permissions 21
Predicated Privileges 21
Deferred Table Creation 22
Online Utilities 22
Merging and Splitting Partitions 22
Maximum Size of Query in the Statement Cache 23
Enhancements to show_cached_plan_in_xml 23
Fast-Logged Bulk Copy 24
Precomputed Result Sets 24
Concurrent dump database and dump transaction Commands 24
Hash-Based Update Statistics 25

Enhancements to dump and load 25
alter table drop column without datacopy 26
Expanded Maximum Database Size 26
User-Defined Optimization Goal 26
Shared Query Plans 27
Initializing Databases Asynchronously 27
In-Row Large Object Compression 27
Configuring Shared Memory Dumps 28
System Changes for Adaptive Server Version 15.7 ESD #2 28
Commands 28
Functions 30
System Stored Procedures 31
Configuration Parameters 32
System Tables 33
Utilities 34
Monitoring Table Changes 35
Version 15.7 ESD #1 37
Proxy Table Support for In-row LOB Columns 37
Support for System Procedures in Replicated Master Database 37
Increased Number of Parameters 37
Multiple Listeners on Windows 37
Windows Supports Adaptive Server High Availability …38
New and Changed Configuration Parameters 38
Version 15.7 39
Application Functionality Configuration Group 39
New Adaptive Server Kernel 39
Compressing Data in Adaptive Server 40
New Security Features 40
End-to-End CIS Kerberos Authentication 41
Dual Control of Encryption Keys and Unattended Startup 41
Securing Logins, Roles, and Password Management Extensions 42

Login Profiles 42
Employee Lifecycle Management 42
External Passwords and Hidden Text 43
Abstract Plans in Cached Statements 43
Shrink Log Space 44
Displaying Currently Set Switches with sysoptions 44
Changes for Large Objects 44
In-Row Off-Row LOB 44
Using Large Object text, unitext, and image Datatypes in Stored Procedures 45
Using LOB Locators in Transact-SQL Statements 45
Extension to where Clause for Large Objects 46
Showing Cached Plans in XML 46
Padding a Character Field Using str 46
Changes to select for update 46
Creating Nonmaterialized, Non-null Columns 47
Sharing Inline Defaults 47
Retain Monitoring Data 47
Analyze Dynamic Parameters 48
Monitor Lock Timeouts 48
Truncate Trailing Zeros 49
Fully Recoverable DDL 49
Transfer Rows from Source to Target Table Using merge 49
View Statistics and Histograms with sp_showoptstats 49
Changes to Cursors 50
Release Cursor Locks at Cursor Close 50
Enhanced Transaction Support for Cursors 50
Monitor Cursor Statements 50
Nested select Statement Enhancements 51
Changes to Commands and System Procedures in Chained Transaction 51
Expanded Variable-Length Rows 52

Changes to like Pattern Matching 53
Changes to Quoted Identifiers 53
Allowing Unicode Noncharacters 53
Reduce Query Processing Latency 54
The sybdiag Utility 54
The Optimizer Diagnostic Utility 55
System Changes for Adaptive Server Version 15.7 55
Commands 56
Functions 59
System Stored Procedures 60
Configuration Parameters 63
System Tables 65
Utilities 67
Monitoring Table Changes 67
Global Variables 74
Version 15.5 Cluster Edition 75
Adaptive Server 15.5 Cluster Edition Feature and Platform Matrix 75
Multiple simultaneous failover 77
Adding space to an archive database 77
Distributed transaction management in the shared-disk cluster 77
System Changes for Adaptive Server Version 15.5 Cluster Edition 78
Changed commands 78
Monitoring Tables 78
Configuration Parameters 80
Functions 80
Version 15.5 81
Adaptive Server 15.5 Feature and Platform Matrix 81
In-Memory and Relaxed-Durability Databases 83
Faster Compression for Backups 84
Backup Server Support for the IBM® Tivoli® Storage Manager 84

Deferred Name Resolution for User-Defined Stored Procedures 85
FIPS 140-2 Login Password Encryption 85
Incremental Data Transfer 86
bigdatetime and bigtime Datatypes 86
Creating and Managing tempdb Groups 87
System Changes for Adaptive Server Version 15.5 87
Datatypes 87
Functions 88
System Stored Procedures 89
Commands 91
Configuration Parameters 93
Monitoring Tables 93
System Tables 93
Utilities 94
Auditing 95
Version 15.0.3 97
SQL Statement Replication 97
Security Enhancements 97
LDAPS User Authentication Enhancement 97
Automatic LDAP User Authentication and Failback 97
Login Mapping of External Authentication 98
Using SSL to Specify a Common Name 98
Concurrent Kerberos Authentication 98
Virtually Hashed Tables 99
Huge Pages 99
Upgrading During a High Availability Configuration 100
Reinstalling System Stored Procedures 100
Distributed Transaction Management (DTM) 100
Adaptive Server Plug-in Updates 101
The Java Interface 102
System Changes for Adaptive Server Version 15.0.3 102
Functions 102

System Stored Procedures 103
Commands 104
Configuration Parameters 104
Monitoring Tables 105
System Tables 105
Version 15.0.2 107
Encrypted Columns 107
Archive Database Access 108
Finding Slow-Running Queries 109
Deferred Compilation 109
Case-Insensitive Sort Orders for Chinese and Japanese Character Sets 110
Statistical Aggregate Functions 110
Standard Deviation and Variance 111
Eager and Lazy Aggregation 111
Vector and Scalar Aggregation 112
Improved Performance for Data Insertion 112
Using Asynchronous Writes During a Page Split 113
Improving Throughput of tempdb Transactions . 113
Post-commit Optimization 114
Changes to the Query Processor 115
Deferred Compilation 115
Non-binary Character Set Histogram Interpolation 115
Expression Histogramming Selectivity Estimates 115
Viewing Current Optimizer Settings 116
New Security Features 116
PAM Support in 64-bit Adaptive Server on AIX . 116
Global Login Triggers Set Automatically 116
SSL Support 117
Improved Password Security 117
Auditing Enhancements 117
High Availability Considerations 118

Installing and Editing Monitoring Tables 118
Monitoring Tables for the Statement Cache 119
Row-Level Locking for System Tables 119
The xmltable() Function 120
Relocated Joins 121
User-Defined SQL Functions 121
instead of Triggers 122
System Changes for Adaptive Server Version 15.0.2 ..122
Trace Flags 123
Commands 123
Changes to the set Command 123
Utilities 124
System Stored Procedures 125
System Tables 128
Configuration Parameters 130
Functions 132
Global Variables 133
Version 15.0.1 135
Changes to Abstract Plans 135
New Query-Level Settings 135
Operator Name Alignment for the Abstract Plan and the Optimizer Criteria 136
Extending the Optimizer Criteria Set Syntax 137
Literal Parameterization 137
System Changes for Adaptive Server Version 15.0.1 ..137
Functions 138
Configuration Parameters 138
Commands 140
Monitoring Tables 142
Version 15.0 143
Partition Support 143
Row-Locked System Catalogs 144
Query Processor 144
Large Identifiers 145
Computed Columns 145

Differences Between Computed Columns and Function-Based Indexes 146
Differences Between Materialized and Not Materialized Computed Columns 146
Scrollable Cursors 146
unitext Datatype Support 147
big int Datatype Support 147
Unsigned Integer Datatype Support 148
Integer Identity 148
Enhancements to XML Services 149
Adaptive Server Plug-in Enhancements 149
Interactive SQL 150
User-Defined Web Services 150
Very Large Storage Support 150
Automatic Running of update statistics 151
SySAM License Management 151
Query Processing Metrics (qp Metrics) 152
Updates to Abstract Plans 153
showplan Changes 153
Secure Socket Layer Uses FIPS 140-2 154
System Changes for Adaptive Server Version 15.0 155
Utilities 155
Reserved Words 155
Global Variables 156
Configuration Parameters 157
Functions 158
Commands 159
System Stored Procedures 161
System Tables 162
Monitoring Tables 166
Obtaining Help and Additional Information 169
Technical Support 169
Downloading Product Updates 169
Product and Component Certifications 170
Accessibility Features 170

Index 173

Version 15.7 SP100

Adaptive Server® version 15.7 SP100 introduces new features and enhancements.

Adaptive Server 15.7 SP100 Feature and Platform Matrix

The feature and platform matrix shows feature availability for supported operating systems in Adaptive Server 15.7 SP100. A “Y” indicates the feature is supported for that platform.

Adaptive Server Options	HP-UX Itanium 64-bit	HP-UX PA Risc 64-bit	IBM AIX 64-bit	Linux on Power 64-bit	Linux Opteron 64-bit	Linux x86 32-bit	Solaris 32-bit	Solaris 64-bit	Solaris Opteron 64-bit	Windows Opteron X64	Windows x86 32-bit
Security and directory services	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Cybersafe Kerberos							Y	Y			Y
Pluggable authentication module	Y		Y	Y	Y	Y	Y	Y	Y		
Fine-grained access control	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
LDAP server directory	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
LDAP user authentication	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Platform Native Kerberos							Y	Y			
Secure Sockets Layer	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
MIT Kerberos Note: Version 4.0.1 for Windows 64	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Encrypted columns, including fine-grained access control	Y	Y	Y		Y	Y	Y	Y	Y	Y	Y
High availability	Y	Y	Y		Y	Y	Y	Y			Y

Adaptive Server Options	HP-UX Itanium 64-bit	HP-UX PA Risc 64-bit	IBM AIX 64-bit	Linux on Power 64-bit	Linux Opteron 64-bit	Linux x86 32-bit	Solaris 32-bit	Solaris 64-bit	Solaris Opteron 64-bit	Windows Opteron X64	Windows x86 32-bit
Partitions	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
In-memory databases	Y	Y	Y	Y	Y			Y	Y	Y	
Tivoli Storage Manager for Backup Server	Y		Y		Y			Y	Y	Y	
Active messaging	Y	Y	Y		Y	Y	Y	Y	Y	Y	Y
Enhanced Full-Text Search (EFTS)		Y	Y		Y	Y	Y	Y			Y
Features included in base Adaptive Server											
Cross-platform dump and load	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Job Scheduler	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Native XML	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
IPv6	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Java	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Web Services	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Distributed transaction management	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Content management (external file support)	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Archived database access	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Shrinking Databases

In Adaptive Server versions 15.7 SP100 and later, use the **alter database** command to shrink databases, freeing unused space for reuse or deletion.

If Adaptive Server encounters data on the portion of the database you are shrinking, it moves the data to a new location before removing the space from the database. Once the portions to be removed are empty, the physical storage is replaced by references to a null device, which frees the space and makes it available for reuse or deletion.

You can shrink databases that are online and in use.

Enhancements to Backup and Restore

Adaptive Server 15.7 SP100 includes enhancements to the **dump database** and **load database** commands.

Enhancements include:

- The **dump database** command now allows you to perform a cumulative backup, in which you make a copy of all the pages that have been modified in the database since the last full database dump.
- The ability to back up (**dump**) one database to restore (**load**) into another, using a dump history file.
- Integration of Tivoli Storage Manager with the dump history file.
- The **sybdumptran** utility, which allows you to generate a final transaction log dump from log devices from outside of Adaptive Server, when your server suffers a catastrophic failure.
- **load database** for **listonly=create_sql** to include database options and attributes.

Aggregating Metrics from Syntactically Similar Queries

Adaptive Server 15.7 SP100 includes the **show_condensed_text** function, which lets you aggregate monitoring data for multiple statements in the statement cache that, although syntactically distinct (having individual comments or differences in their text format), represent semantically identical queries.

By providing a way to identify semantically equivalent statements, **show_condensed_text** allows you to aggregate the values from `monCachedStatement` into a single row in the query result.

Updates to Precomputed Result Sets

Updates to the precomputed result sets include grant and revoke access permissions for precomputed result sets created with the **immediate refresh** parameter for the **select**, **update statistics**, and **delete statistics** commands. Also, **sp_showoptstats** and **sp_depends** display additional information about precomputed result sets.

Improved Data Load Performance

Adaptive Server 15.7 SP100 improves data load performance by enabling **ins_by_bulk** optimization criteria, which uses bulk-data row inserts for **insert** statements.

When you enable data load optimization criteria using the **set** statement, Adaptive Server then uses that criteria when running subsequent **insert...select** or batch insert statements. The scope of the statement-level directive enabled via the abstract plan is limited to the statement.

This optimization improves data load performance on the target table.

Incremental Reorganization

Adaptive Server 15.7 SP100 includes a new parameter—**defrag**—for the **reorg** command that lets you schedule and resume reorganization, and allows concurrent reads or writes on the data being reorganized.

reorg defrag locks each row or page, as per the locking scheme of the object, and begins and commits a transaction for every data chunk that is processed. The data reorganization space requirement does not exceed the size of an allocation unit (256 data pages). Index updates for reorganized data do not consume extra space. For every data partition undergoing incremental reorganization, a row is stored in `sysattributes`.

Creating Indexes Without Blocking Access to Data

Adaptive Server 15.7 SP100 includes the **create index ... online** parameter, which allows you to create indexes without blocking access to the data you are indexing

Query Plan and Execution Statistics in HTML

In Adaptive Server versions 15.7 SP100 and later, you can generate a graphical query plan in HTML format that you can view in a Web browser.

To generate a graphical representation of query plan execution statistics, use the HTML **set statistics** commands when you execute a query plan. These HTML representations provide a description of a query plan that includes a tree structure and a timing diagram on a single page.

Gathering Hash-Based Statistics with create index

Adaptive Server 15.7 SP100 extends the support of gathering hash-based statistics to the **create index** command.

The **create index** hash-based statistics gathering options let you gather minor index attributes while an index is being created, either serially or in parallel. This saves you from having to update statistics, with regard to minor index attributes, after creating the index.

Query Plan Optimization with Bloom Filters

Adaptive Server 15.7 SP100 introduces bloom filters, which optimize query plans by improving join performance.

A bloom filter provides early filtering of rows that cannot be joined before they reach the join operator. In Adaptive Server, bloom filters are implemented for hash join, sort-merge joins, and reformatting-based nested-loop joins. They are applicable only to equi-joins, and chosen by the optimizer for the final plan based on costing.

Support for Replication by Column Value

Adaptive Server 15.7 SP100 extends Multi-Path Replication™ by including a new distribution model that provides support for distributing replicated rows based on the data values in one or more columns in a single table.

Support for Multiple Scanner Threads

The Adaptive Server monitoring tables support multiple scanner threads for RepAgent.

Sybase Central Adaptive Server Plug-in

Sybase® no longer maintains the Sybase Central Adaptive Server plug-in. In Adaptive Server versions 15.7 SP100 and later, use Sybase Control Center for Adaptive Server.

Running Adaptive Server on VMware vSphere

Adaptive Server version 15.7 SP100 supports VMware vSphere.

Support for OData

Adaptive Server version 15.7 SP100 supports SAP® Sybase OData Server which enables data services over RESTful HTTP, allowing you to perform operations through URIs (Universal Resource Identifiers) to access and modify information.

Support for NTLM and MIT Kerberos on Windows 64-bit

MIT Kerberos and NT Lan Manager security services are supported on Windows 64-bit.

Properties Added to Responses File

New properties have been added to the Adaptive Server responses file for all platforms.

These properties have been added to the responses file:

- DO_UPDATE_ASE_SERVER
- UPDATE_ASE_SERVER_ NAME_[n]
- UPDATE_ASE_PASSWORD_[n]

Automatic Physical Database Rearrangement

When loading a dump of a database that had a segregated log and data segment, Adaptive Server 15.7 SP100 and later will now rearrange the physical layout of the target database to ensure physical separation of the log and data segments.

System Changes for Adaptive Server Version 15.7 SP100

Adaptive Server 15.7 SP100 includes changes to commands, functions, system procedures, configuration parameters, system tables, and monitoring tables.

Commands

Adaptive Server 15.7 SP100 contains changed commands.

Table 1. Changed commands

Command	Description
alter database	Adds the **off** parameter, which specifies the device names from which you are releasing space.
create index	• The **online** parameter lets you create indexes without blocking access to the data you are indexing. • You may execute parallel **create index** for a clustered index on a data-only-locked table using the **sorted_data** parameter. • Use the **with statistics** hash-based option and additional hash option clauses to gather index statistics on tables.
create materialized view	You cannot include a **like** clause with **create materialized view** that includes an **immediate refresh** parameter.
create precomputed result sets	You cannot include a **like** clause with **create precomputed result set** that includes an **immediate refresh** parameter.
dbcc	For every data partition undergoing incremental reorganization, a row is stored in `sysattributes`. Use the new **dbcc** parameter **zapdefraginfo** to delete this information in `sysattributes` before performing a downgrade.
dump database	A new **cumulative** allows you to specify a backup as a cumulative incremental dump.
load database	A new **cumulative** type allows you to load a backup created with the **dump database cumulative** keyword.
reorg	The **defrag** parameter lets you to schedule and resume reorganization while also allowing concurrent reads or writes on the data being reorganized.

Command	Description
select	The **ins_by_bulk** parameter improves performance by directly inserting data rows into newly allocated data pages by bulk for tables. Set the parameter at the query level using the abstract plan for a specific insert statement.
set **ins_by_bulk**	The **set** command enables the new **ins_by_bulk** parameter for the session, which improves performance by directly inserting data rows into newly allocated data pages for tables.
set **join_bloom_filter**	Enable or disable the use of bloom filters for query plan optimization.
set **statistics** • **plan_detail_html** • **parallel_ plan_detail_html** • **plan_directory_html**	• Generate a graphical query plan in HTML format • Generate a graphical query plan in HTML format containing execution statistics related to the timing spent in each operator per execution phase • Generate a graphical query plan in HTML format containing information details of plan operators such as the name, different timestamps captured during the execution, number of rows affected, number of estimated rows, elapsed time • Generate a graphical query plan in HTML format containing information details per thread and plan fragments for query plans that are executed in parallel using several worked threads. • Specify the directory path name into which to write the HTML query plans.
update statistics	**update statistics** acquires memory from the default data cache instead of `tempdb` buffer cache if the session is using a `tempdb` bound to an inmemory device.

Functions

Adaptive Server 15.7 SP100 contains new functions.

Table 2. New Functions

Command	Description
defrag_status	Returns metrics of any defragmentation operation that is started or ongoing on the named object or partition.
loginfo	Returns information about a transaction log.

Command	Description
show_condensed_text	Lets you aggregate monitoring data for multiple statements in the statement cache that, although syntactically distinct (having individual comments or differences in their text format), represent semantically identical queries.
shrinkdb_status	Determines the status of a shrink operation.

System Stored Procedures

Adaptive Server 15.7 SP100 contains new and changed system stored procedures.

Table 3. New System Stored Procedures

System Stored Procedures	Description
sp_helpdefrag	Reports defragmentation information for either all eligible objects for **reorg defrag** in the database from which the command is invoked from or for the given object if it is eligible for **reorg defrag**.
sp_dump_info	**sp_dump_info** displays the size of data and log that a cumulative dump would contain at a specific point in time.

Table 4. Changed System Stored Procedures

System Stored Procedures	Description
sp_configure	• Enable concurrent transaction dumps and cumulative database dumps by setting the **enable concurrent dump tran** parameter to 1. • Enables the gathering of index attributes using hash-based statistics when creating an index by including the **utility statistics hashing** parameter.
sp_config_rep_agent	Supports the additional distribution model **filter** for the **multipath distribution model** property.
sp_dboption	Includes the **allow incremental dumps** parameter which supports the backup and restoration of cumulative dumps.

System Stored Procedures	Description
sp_depends	Displays additional information about precomputed result sets.
sp_dump_history	Supports the same functionality for cumulative dumps as for full database dumps when you specify the **cumulative** keyword in the *@dump_type* parameter.
sp_extrapwdchecks	Supports NULL values for **caller_password** and **loginame** parameters.
sp_helpdb	Output includes information about the durability of a user-created temporary database.
sp_helprotect	Includes the owner name in the `object` column for objects (tables, databases, and so on).
sp_listener	Adds support for the AF_UNIX protocol.
sp_optgoal	When granular permissions is enabled you must be a user with `manage opt goal` to create or delete a goal.
sp_replication_path	Supports binding, unbinding, and listing of filters.
sp_showoptstats	Displays additional information about precomputed result sets.
sp_sjobcreate	Includes the **continuous_run** parameter, which allows you to execute jobs from a starting time and run them according to specified intervals until an end time.
sp_sjobmodify	Includes the **continuous_run** parameter, which allows you to execute jobs from a starting time and run them according to specified intervals until an end time.

Configuration Parameters

Adaptive Server 15.7 SP100 introduces new and changed configuration parameters.

New Configuration Parameters

Configuration Parameter	Description
enable ISM	Enables and disables Adaptive Server to use integrated service management (ISM) on the Solaris platform.
enable HugePages	Enables and disables Adaptive Server to use huge pages on Linux platforms that support huge pages.
max util parallel degree	Specifies the server-wide maximum number of worker processes allowed per query used by the **create index with consumers** and **update stats with consumers** commands.
optimize temp table resolution	
solaris async i/o mode	Allows you to select various asynchronous IO modes on the Solaris platform.

Changed Configuration Parameters

Configuration Parameter	Description
enable permissive unicode	Allows you to include random binary data when enabled.
permission cache entries	Determines the number of cache protectors per task, increasing the amount of memory for each user connection and worker process.

System Tables

Adaptive Server 15.7 SP100 contains new and changed system tables.

New System Tables

Table	Column
`sysdams`	Stores the dump allocation map (DAM) for the database.

Changed System Tables

Table	Column
`sysloginroles`	Adds the `predid` column, which is the object ID for the predicate of a **grant role** command.

Utilities

Adaptive Server 15.7 SP100 contains new and changed utilities.

New Utilities

Command	Description
ddlgen	An *Xextended_object_type*, –XDE has been added for -TDB. The new type allows you to generate a database and all of its objects in correct dependent order.
optdiag	**optdiag** adds the **prs_name** parameter.
sybdumptran	Generates a transaction log dump from the log pages contained in operating system files/raw devices, that were formerly used as a log device by a database in an Adaptive Server environment.

Changed Utilities

Command	Description
Backup Server	Supports **mountwait -M** parameter for the **dump database**, **dump transaction**, **load database**, and **load transaction** commands.
optdiag *prs_name*	Indicates the name of a precomputed result set.

Monitoring Table Changes

Adaptive Server version 15.7 SP100 includes new and changed monitoring tables.

New Monitoring Tables

Monitoring Table	Description
`monRepCoordinator`	Provides information on the RAT coordinator process when RAT runs in Multiple Scanner mode.
`monRepSchemaCache`	Reports the schema cache information for each scanner running on a replication server.

Monitoring Table	Description
monSysExecutionTime	Provides execution time, in microseconds, of each operation performed.

Changed Monitoring Tables

Monitoring Table	Description
monOpenObjectActivity	Adds the columns NumLevel0Waiters and AvgLevel0WaitTime.
monRepLogActivity	The MaxHashSchemaSize and NumberOfSchemasReused columns of the monRepLogActivity table have been removed.
monRepScanners	The NumberOfTruncPointRequested and NumberOfTruncPointMoved columns are populated only when the single task scanning of a log model is used.
monRepSenders	Adds these columns: • NumberOfTruncPointRequested • NumberOfTruncPointMoved • AvgTruncPointInterval
monSpinlockActivity	Adds the SpinlockSlotID column

Monitoring Table	Description
monState	Adds these columns: • TableAccesses • IndexAccesses • ULCFlushes • ULCFlushFull • WorkTables • TempDBObjects • Rollbacks • Selects • Updates • Inserts • Deletes • Merges • ULCKBWritten • PagesRead • PagesWrite • PhysicalReads • PhysicalWrites • LogicalReads

Permission Changes for Commands and Functions

Adaptive Server 15.7 SP100 includes permission changes for commands and functions.

Changed Permissions

Configuration parameter	Description
asehostname	With granular permissions enabled, to query the host name on which the dataserver is running, you must have `manage server` privileges, or have been granted `select` permission on **asehostname**. With granular permissions disabled, you must be a user with sa_role or have `select` permission on **asehostname**.

Configuration parameter	Description
dbcc pravailabletempdbs	With granular permissions enabled, you must be a user with `manage server` privilege to execute **pravailabletempdbs**. With granular permissions disabled, you must be a user with sa_role.
dbcc serverlimits	With granular permissions enabled, you must be a user with `manage server` privilege to execute **serverlimits**. With granular permissions disabled, you must be a user with sa_role.
dbcc cis showcaps	With granular permissions enabled, you must be a user with `manage server` privilege to execute **cis showcaps**. With granular permissions disabled, you must be a user with sa_role.
dbcc cis remcon	With granular permissions enabled, you must be a user with `manage server` privilege to execute **cis remcon**. With granular permissions disabled, you must be a user with sa_role.
sp_optgoal	A new server level privilege, `manage opt goal` enables users with sa_role and sa_serverprivs_role to write or delete goals using **sp_optgoal**.

Version 15.7 ESD #4

Adaptive Server version 15.7 ESD #4 introduces new features and enhancements.

15.7 ESD #4.2

New features in Adaptive Server Enterprise 15.7 ESD #4.2.

Improved Rollback Reporting

Adaptive Server 15.7 ESD #4.2 onwards provides enhanced reporting using the **kill with statusonly** command.

15.7 ESD #4

New features in Adaptive Server Enterprise 15.7 ESD #4.

Remote Backup Server Maximum Name Length

The remote backup server name length limit for **dump** and **load** commands has been increased from 30 characters to 255 characters.

Support for Query Plan Pinning

The configuration option dynamic SQL plan pinning has been introduced in Adaptive Server 15.7 ESD #4 to improve Adaptive Server performance by reducing the time spent by server connections waiting for access to the query plan manager.

AF_UNIX Support

Adaptive Server versions 15.7 SP100 and later support the AF_UNIX named pipe entries in the interfaces file, which avoid some of the overhead the TCP protocol incurs.

To create AF_UNIX entries, use the **sp_listener** system procedure.

Changes to sp_addthreshold and Roles

System and user defined roles that are active when a threshold procedure is created are preserved in `systhresholds`.

System Changes for Adaptive Server Version 15.7 ESD #4

Adaptive Server 15.7 ESD #4 includes changes to utilities and configuration parameters.

Utilities

Backup Server now supports the Tivoli Storage Manager **mountwait -M** parameter for the **dump database**, **dump transaction**, **load database**, and **load transaction** commands.

Configuration Parameters

Adaptive Server 15.7 ESD #4 introduces two new configuration options.

Configuration Parameter	Description
early row send increment	Configures the additional number of rows sent in the second and subsequent packets of a result set (subject to the maximum packet size).
number of early send rows	Configures the number of rows Adaptive Server sends to the client in the first packet of a new result set.

Version 15.7 ESD #3

Adaptive Server version 15.7 ESD #3 introduces new features and enhancements.

sybrestore Utility

sybrestore restores an Adaptive Server database to the time of failure from the most current full database backup dump files.

See the *Adaptive Server Utility Guide* for ESD #3.

System Changes for Adaptive Server Version 15.7 ESD #3

Adaptive Server 15.7 ESD #3 includes changes to functions and commands.

Commands

Adaptive Server 15.7 ESD #3 contains changed commands.

Table 5. Changed commands

Command	Description
writetext	**writetext** logs operations on IRL columns. You can run the **writetext** command (with or without the **with log** parameter) simultaneously with the **online** parameter.

Functions

Adaptive Server 15.7 ESD #3 contains new functions.

Table 6. New Functions

Command	Description
allocinfo	Returns a list of allocation pages that are stored in an object allocation map (OAM) page.
spaceusage	Returns metrics as a comma-separated string for space use in Adaptive Server.

System Stored Procedures

Adaptive Server 15.7 ESD #3 contains new and changed system stored procedures.

Table 7. Changed System Stored Procedures

System Stored Procedures	Description
sp_helpconfig	The **estimate** parameter determines the approximate amount of memory required for certain memory-intensive configuration parameters.
sp_dboption	The **deallocate first text page** options has been added to sp_dboption.

Monitoring Table Changes

Adaptive Server version 15.7 ESD #3 includes a new monitoring table.

Monitoring Table	Description
`monMemoryUsage`	Provides information about server and kernel memory pools, including metrics about their sizes, usage patterns, and availablility.

Version 15.7 ESD #2

Adaptive Server® version 15.7 ESD #2 introduces many new features and enhancements.

Granular Permissions

Granular permissions enable you to grant system privileges, allowing you to construct site-specific roles with privileges to match your requirements, and restrict system administrators and database owners from accessing user data.

Grantable system privileges are granular and allow you to enforce principles of "separation of duties" (which requires that, for particular sets of operations, no single individual be allowed to execute all operations within the set) and "least privilege" (which requires that all users in an information system should run with as few privileges as are required to do the job).

All granted privileges are immutable. That is, you cannot revoke or grant one privilege from —or to—another privilege. However, privileges may overlap what the grantee can do. Possessing one privilege may imply possessing another, more granular, privilege.

Enabling granular permissions reconstructs system-defined roles (sa_role, sso_role, oper_role, and replication_role) as privilege containers consisting of a set of explicitly granted privileges. You may revoke explicitly granted system privileges in system-defined roles and regranted from the roles.

See "Using Granular Permissions" in the *Security Administration Guide* for information about using and configuring Adaptive Server with granular permissions. See the Reference Manual: Commands and the Reference Manual: Procedures to see how enabling Adaptive Server for granular permissions affects individual commands and system procedures. .

Predicated Privileges

Predicated privileges provide a system of flexible row-level access controls, allowing you to grant, select, update, and delete privileges to different users, groups, or roles based on a predicate Adaptive Server evaluates when it accesses the data. If the condition expressed by the predicate is not met for any row of data, Adaptive Server withholds that row from the result set.

Predicated privileges offer data privacy protection based on relieve access controls that dynamically grant privileges to a user based on data content or context information, allowing you to implement a privacy policy in the server instead of the client or a Web server.

A predicate may access other objects, such as tables, SQL functions, or built-in functions. These accesses are checked against the permissions and roles of the predicate owner (such as the grantor) instead of requiring explicit permission by the user who executes the **select**, **update**, or **delete** command on the objects accessed by the predicate.

Predicated privileges allow a service provider to store data in a single database, and share the same tables for multiple customers instead of requiring separate views and instead of triggers for each customer.

See "Granting Predicated Privileges" in the *Security Administration Guide* for information about using and configuring Adaptive Server with predicated privileges.

Deferred Table Creation

create table...with deferred_allocation allows you defer the page allocation for a table.

The **with deferred_allocation** parameter for the **create table** command lets you defer page allocation for a table. Deferred tables help applications that create numerous tables, but use only a small number of them. Tables are called "deferred" until Adaptive Server allocates their pages.

System tables include entries for deferred tables. These entries allow you to create objects associated with deferred tables such as views, procedures, triggers, and so on..

Adaptive Server performs page allocation for deferred tables when it inserts the first row (called table materialization). Access to the table before the first **insert**, such as selects, deletes or updates, functions that report space usage, or enforce referential integrity constraints during DML on other tables, behave as if the table is empty. That is, a **select** against a deferred table produces an empty result set. Although you can create indexes on deferred tables, the page allocation for these indexes is deferred until Adaptive Server materializes the table.

Online Utilities

Adaptive Server versions 15.7 ESD # 1 and later include an **online** parameter for **reorg rebuild** that lets you reorganize data and perform maintenance on tables without blocking users data from users.

Merging and Splitting Partitions

Over time, a partition's data distribution may become skewed, or the manner in which the data was originally partitioned may not suit current business requirements. Use **alter table** to

merge, split, or move partitions to redistribute the data and revive the performance benefits of using partitions.

For example:

- Splitting partitions – a company divides data into four partitions according to regions — North, South, East and West— so customer representatives have fast and efficient access to their regions' customers, independent of other regions. If sales increase in the Southern region and the customer base has expanded significantly, frequent queries involving partition scans and maintenance operations may cause the South partition to be slow and inefficient, losing out on the benefits of partitioning the customer data. In this situation, splitting the data in the South partition into two partitions, South-East and South-West, may revive performance without affecting the data in other partitions.
- Merging partitions – a company's sales data is partitioned into the four yearly quarters— Q1, Q2, Q3, and Q4. At the end of the year, the company merges the data for the year and archives it. Merging partitions that represent a closed financial year is efficient because sales' data for a past year is accessed infrequently, and the older data is most likely to be read but not updated.

Maximum Size of Query in the Statement Cache

Adaptive Server versions 15.7 ESD #2 and later allow you to store very large SQL statements. You can save individual statements of up to 2MB (for a 64-bit machine) in the statement cache.

Versions of Adaptive Server earlier than 15.7 ESD #2 had a 16K limit for individual statements stored in the statement cache, even if statement cache size was configured with a larger size.

Enhancements to show_cached_plan_in_xml

Adaptive Server versions 15.7 ESD #2 and later include new information for **show_cached_plan_in_xml**.

show_cached_plan_in_xml includes output for:

- Scan coverage
- Worktables
- Dynamic partition elimination
- Total logical I/O (lio) and total physical I/O (pio)

Fast-Logged Bulk Copy

Adaptive Server version 15.7 ESD #2 and later allows you to fully log **bcp** in fast mode, which provides faster data throughput and full data recovery. Earlier versions logged only page allocations.

Use the **set logbulkcopy {on | off }** command to configure fast-logged **bcp** for the session. You may include the **set logbulkcopy {on | off }** with the **--initstring** *'Transact-SQL_command'* parameter, which sends Transact-SQL commands to Adaptive Server before transferring the data. For example, to enable logging when you transfer the `titles.txt` data into the `pubs2..titles` table, enter:

```
bcp pubs2..titles in titles.txt --initstring 'set logbulkcopy on'
```

You must enable **select into/bulkcopy/pllsort** on the database before issuing **fast**-logged **bcp**; otherwise, **bcp** uses slow mode.

Precomputed Result Sets

Adaptive Server versions 15.7 ESD #2 and later allow you to create precomputed result sets.

A precomputed result set is a view for which the result is computed, stored, and available for future use. Once configured for precomputed result sets, Adaptive Server precomputes a query and attempts to use the precomputed result to answer the actual query. Precomputed result sets are also called materialized views.

Conceptually, a precomputed result set is both a view (because it includes query definition stored in the system tables) and a table (because it includes persistent data). You can run many of the same operations that you perform on tables on precomputed result sets as well, including creating indexes and running update statistics.

Use the **create**, **alter**, and **refresh** commands on precomputed result sets.

Concurrent dump database and dump transaction Commands

Adaptive Server versions 15.7 ESD #2 and later allow a **dump transaction** command to run concurrently with a **dump database** command, reducing the risk of losing database updates for a longer period than that established by the dump policy.

Hash-Based Update Statistics

Adaptive Server versions 15.7 ESD #2 and later allow you to gather hash-based statistics on minor index attributes and unindexed columns instead of using sort-based statistics, significantly reducing elapsed time and resource usage. Using hash-based statistics improves performance by reducing the number of required scans, and avoiding disk-based sorting.

Hash-based statistic allow greater flexibility than sort-based statistics:

- Running hash-based statistics should require less time, increasing the amount you can accomplish during a maintenance window.
- Because hash-based statistics require less procedure cache, you may be able to run **update statistics** on a data-only-locked table outside a maintenance window, since the Adaptive Server `tempdb` buffer cache (which typically uses the default data cache) is typically much larger than the procedure cache, reducing the impact of **update statistics**.
- Hash-based statistics do not generally require large `tempdb` disk allocations. If you previously increased the size of `tempdb` to accommodate large sorts from `update statistics`, you may be able to redeploy this space.
- **update [index | all] statistics** with hashing may run faster than **update [index | all] statistics** with sampling. However, an exception may be **update statistics** *table_name*(*col_name*).
- **update statistics table_name** *(col_name1), (col_name2) . . .* with hashing allows you to collect statistics on several columns with a single scan instead of several scans.

Enhancements to dump and load

Adaptive Server 15.7 ESD #2 includes enhancements to the **dump** and **load** commands, which make it easier for you to back up and restore your databases.

The enhancements include:

- The **dump configuration** command allows you to back up the Adaptive Server configuration file, the dump history file, and the cluster configuration file.
- Dump configurations define options to create a database dump. Backup Server then uses the configuration to perform a database dump. You can use:
  - The dump configuration to create, modify, or list dump configurations, then use **dump database** or **dump transaction** with the configuration.
  - The **enforce dump configuration** configuration parameter to enable dump operations to use a dump configuration.
  - The configuration group "dump configuration," which represents user-created dump configurations.

- Dump history:
  - Preserve the history of **dump database** and **dump transaction** commands in a dump history file that Adaptive Server can later use to restore databases, up to a specified point in time.
  - Read the dump history file and regenerate the load sequence of SQL statements necessary to restore the database.
  - Use **sp_dump_history** to purge dump history records.
  - Use the **enable dump history** configuration parameter to disable default updates to the dump history file at the end of every dump operation.
  - Use the **dump history update** configuration parameter to specify the name of the dump history file.
- Dump header – New options to the **dump with listonly** command:
  - **create_sql** – lists the sequence of **disk init**, **sp_cacheconfig**, **create database**, and **alter database** commands required to create a target database with the same layout as the source database.
  - **load_sql** – uses the dump history file to generate a list of **load database** and **load transaction** commands required to repopulate the database to a specified point in time.

alter table drop column without datacopy

Adaptive Server versions 15.7 ESD #2 and later add the **no datacopy** parameter to the **alter table ... drop column** command, which allows you to drop columns from a table without performing a data copy, reducing the amount of time required for **alter table ... drop column** to run.

Expanded Maximum Database Size

Adaptive Server versions 15.7 ESD #2 and later expand the maximum size of a database to approximately 64 terabytes by converting the logical page number from a signed integer to an unsigned integer.

Versions of Adaptive Server earlier than 15.7 ESD #2 allowed for a maximum database size of approximately 32 terabytes.

User-Defined Optimization Goal

Adaptive Server versions 15.7, ESD #2 and later allow you to create user-defined optimization goals.

User-defined optimization goals allow you to:

- Create a new optimizer goal

- Define set of active criteria
- Activate the goal at the server, session, procedure, and query level
- Dynamically change the goal content, without disconnecting and reconnecting the client session

Once you create the user-defined optimization goals, you can invoke them at the server level or for a user session.

Shared Query Plans

Adaptive Server versions 15.7 ESD #2 and later allow you to share query plans, which are cloned from primary query plans, avoiding the need for Adaptive Server to create or recompile query plans that are identical to existing plans.

You should see a performance improvement as Adaptive Server shares query plans instead of reusing or recompiling them. You may see a slight change to procedure cache memory usage as primary query plans are pinned in the cache while Adaptive Server uses their shared query plans.

Initializing Databases Asynchronously

Adaptive Server versions 15.7 ESD #2 and later include the **async_init** parameter for the **alter database** and **create database** commands, which allows you to asynchronously initialize a database while it is being used.

The database initialization is transparent to the user: the database is immediately available when it is created or altered, not when the database initialization is complete.

Any task that uses a page of the database that is not yet initialized performs an initialization of the allocation unit on which the page resides.

The asynchronous initialization is performed by a service task that is started by the **create database** or **alter database** commands. When it restarts, Adaptive Server automatically starts a new service task that completes the initialization. In a clustered environment, if an instance running the service task fails or is shut down, the coordinating instance starts a new service task to complete the initialization.

In-Row Large Object Compression

Adaptive Server versions 15.7 ESD #2 and later support in-row large object (LOB) compression.

Configuring Shared Memory Dumps

Adaptive Server versions 15.7 ESD #2 and later allow you to automatically perform compressed, shared memory dumps according to specific, configurable conditions.

Use the **memory dump compression level** configuration parameter to set the amount of compression Adaptive Server performs for shared memory dumps. Use **sp_shmdumpconfig** to configure the shared memory dumps.

System Changes for Adaptive Server Version 15.7 ESD #2

Adaptive Server 15.7 ESD #2 includes changes to commands, functions, system procedures, configuration parameters, system tables, and monitoring tables.

Commands

Adaptive Server 15.7 ESD #2 contains new and changed commands.

Table 8. New commands

Command	Description
create {precomputed result set | materialized view }	Defines precomputed result sets.
alter {precomputed result set | materialized view }	Alters the properties or policies of a precomputed result set.
refresh {precomputed result set | materialized view }	Refreshes the specified precomputed result set.
drop {precomputed result set | materialized view }	Drops a precomputed result set.
truncate {precomputed result set | materialized view }	Truncates the data in a precomputed result set.
dump configuration	Creates a backup of the Adaptive Server configuration files into a specified dump directory. The copy is created by the Adaptive Server, not the Backup Server.

Table 9. Changed Commands

Command	Change
alter database	**noasync_init** – Indicates that you are extending a database, and that Adaptive Server initializes the extended space asynchronously
alter table	• **with immediate_allocation** – creates regular, nondeferred tables. • **split partition** – redistributes data to two or more partitions. • **merge partition** – combines the data from two or more merge-compatible partitions into a single partition. • **move partition** – moves a partition (and its index) to a specified segment. • **drop column** – drops columns from a table without performing a data copy. • **noasync_init** – indicates the database is initialized synchronously.
create database	• **noasync_init** – indicates the database is initialized synchronously.
create index	Allows you to issue a parallel form of **create index** that uses the query execution engine to more efficiently execute the command.
create table	• **with deferred_allocation** – creates deferred tables. • **with immediate_allocation** – creates regular, nondeferred tables.
dump database	Dumps a database according to the settings in the dump configuration file.
grant	• **as** *pred_name* – the name of the predicate • **grantby** *grantor* – indicates the grantor who grants permission to the user or role.
grant role	**where** *pred_expression* – The SQL condition that must be satisfied when the named role is activated.
load database	New parameters: • **listonly=load_sql** – generates a sequence of **load database** and **load transaction** SQL statements to restore a database to a specified point in time. • **until_time** – the database is restored up to this date and time. • **listonly=create_sql** – generates a sequence of **create database** and **alter database** SQL statements.

Command	Change
load transaction	**listonly=create_sql** – generates a sequence of **create database** and **alter database** SQL statements.
merge	**merge** target tables may include triggers.
reorg rebuild	**with online** – allows you to reorganize your data without taking it offline
set	• **materialized_view_optimization** – determines which precomputed result sets are considered during query optimization • **mon_stateful_history** – when disabled, queries to the historical monitoring tables return all rows in the table buffer. When enabled, queries to the historical monitoring tables return only rows that were added to the tables since **mon_stateful_history** was disabled. • **show_transformed_sql** – displays the SQL text for statements during the Adaptive Server preprocessing phase
update statistics	• **no_hashing** – (the default) uses the sort-hashing algorithm from versions of Adaptive Server earlier than 15.7 ESD #2. • **partial_hashing** – Adaptive Server uses hashing for low unique count domains.

update index statistics, **update statistics**, and **update all statistics** include the **print_progress** parameter, which allows these commands to display progress messages.

See the *Reference Manual: Commands*.

Functions

Adaptive Server 15.7, ESD #2 contains new and changed functions.

Changed Functions

These functions now return an `unsigned` result instead of an `int`:

- **curunreservedpgs** (the **lstart** and **unreservedpgs** parameters also return an unsigned int)
- **used_pages**
- **data_pages**
- **reserved_pages**
- **lct_admin**

New Functions

Adaptive Server 15.7, ESD #2 adds these functions:

- **show_cached_text** – displays the SQL text of a cached statement
- **show_cached_text_long** – displays the SQL text for cached statements longer than 16K

System Stored Procedures

Adaptive Server 15.7, ESD #2 contains new and changed system procedures.

Table 10. New System Stored Procedures

System Stored Procedures	Description
sp_config_dump	Allows you to list, add, or change dump configurations.
sp_dump_history	Allows you to purge dump records from the dump history file. The original dump history file is saved with the timestamp suffixed to the file name.
sp_optgoal 'show','goal_name'	Reports all individual criteria activated by the user-created optimizer goal.
sp_restore_system_role	Restores the system-defined roles or database owner privileges to the system defaults.
sp_shmdumpconfig	Configures shared memory dumps.

Table 11. Changed System Stored Procedures

System Stored Procedures	Description
sp_dboption	Adds the deferred table allocation parameter to configure the database so Adaptive Server defers page allocation for all subsequently created user tables.
sp_checksource	Allows you to include a predicated privilege for *objname*
sp_hidetext	Allows you to include a predicated privilege for *objname*

These system procedures display information about predicated privileges:

- **sp_helprotect**
- **sp_help**
- **sp_helptext**
- **sp_checksource**

These system procedures display information about granular permissions:

- **sp_help**
- **sp_helprotect**

See the *Reference Manual: Procedures*.

Configuration Parameters

Adaptive Server 15.7 introduces new and changed configuration parameters.

New Configuration parameter	Description
enable concurrent dump tran	Enables or disables concurrent dumps.
enable predicated privileges	Enables Adaptive Server to use predicated privileges
update statistics hashing	Enables Adaptive Server to gather hash-based statistics.
enforce dump configuration	Determines if Adaptive Server uses a dump configuration.
enable dump history	Enables and disables updates to the dump history file at the end of database dump.
dump history filename	Specifies the path of your dump history file.
enable plan sharing	Use shared query plans.
enable async database init	Ensures that all **create database** and **alter database** commands initialize databases asynchronously.
memory dump compression level	Controls the compression level for shared memory dumps.

Adaptive Server version 15.7 ESD #2 changes the required role from system administrator to system security officer for these configuration parameters:

- **allow updates to system tables**
- **SQL Perfmon Integration**
- **syb_sendmsg port number**

System Tables

Adaptive Server 15.7 ESD #2 contains changed system tables.

Table 12. Changed System Tables

System Table	Column Added	Description
`sysattributes`	• `object_cinfo2` • `object_date-time`	• Provides a character description for the object • Provides the date and time for the object • The **SP** *object_type* stores options related to RSA key-pair regeneration and LR *object_type*, which stores options related to login profiles
`sysobjects`	`type`	Adds the • `RS` – indicates a precomputed result set • PP – indicates the predicate of a privilege

`sysprotects`	• `pred_id`
	• `protstatus`
		• Object ID of predicated privileg
		• One of:
		  • PROT_PREDICATED – indicates that the privilege (or denial) is predicated
		  • PROT_ROW_FILTER – indicates that the predicate is a **where** clause
		Object ID of predicated privileg

The datatypes for these columns in these tables have changed from `int` to `unsigned int`:

Table 13. Columns That Changed From in To unsigned int

Table	Column
sysusages	• lstart • size • unreservedpgs
sysaltusages	• lstart • size
syspartitions	• firstpage • rootpage • dataoampage • indoampage
systabstats	• leafcnt • pagecnt • emptypgcnt • warmcachepgcnt • unusedcnt • oampgct
syslocks	page
syslogshold	page
systhresholds	free_space

Utilities

Adaptive Server 15.7 contains these changed utilities.

Command	Description
bcp	Adaptive Server version 15.7, ESD #2 and later allows you to fully log fast **bcp**, providing full data recovery.
optdiag	The output from the 15.7 ESD #2 version of **optdiag** cannot be read by earlier versions of **optdiag**. Use the **optdiag -T1** flag with earlier versions of **optdiag** to create output files these versions can read.

Command	Description
dataserver	• **-A** *system_role* – when enable granular permissions is set to 0, and all users are unable to log into Adaptive Server, provides the server administrator with a login account with sso_role • **-n system_privileges** – when `enable granular permissions` is set to 1, and all users are unable to log into Adaptive Server, provides the server administrator with a login account with change password privilege

Monitoring Table Changes

Adaptive Server version 15.7 ESD #2 includes changes to some monitoring tables.

`monCachedStatement` adds these columns:

Monitoring table	Description
`AvgScanRows`	Average number of scanned rows read per execution
`MaxScanRows`	Maximum number of scanned rows read per execution
`AvgQualifyingReadRows`	Average number of qualifying data rows per **read** command execution
`MaxQualifyingReadRows`	Maximum number of qualifying data rows per **read** command execution
`AvgQualifyingWriteRows`	Average number of qualifying data rows per **write** command execution
`MaxQualifyingWriteRows`	Maximum number of qualifying data rows per **write** command execution
`LockWaits`	Total number of lock waits
`LockWaitTime`	Total amount of time, in milliseconds, spent waiting for locks
`SortCount`	Total number of sort operations
`SortSpilledCount`	Total number of sort operations spilled to disk
`TotalSortTime`	Total amount of time, in milliseconds, spent in sorts

Monitoring table	Description
MaxSortTime	Maximum amount of time, in milliseconds, spent in a sort

Version 15.7 ESD #1

Adaptive Server version 15.7 ESD #1 introduces many new features and enhancements.

Proxy Table Support for In-row LOB Columns

Use Adaptive Server® version 15.7 ESD #1 to create in-row large object (LOB) columns. However, because proxy tables do not store information about remote in-row LOB columns in their metadata, when information about the LOB column on the source or target table is unavailable, Adaptive Server stores data off-row on the target table.

Support for System Procedures in Replicated Master Database

Adaptive Server 15.7 ESD #1 allows replication for these system procedures in a replicated master database:

- **sp_addexternlogin**
- **sp_dropexternlogin**
- **sp_maplogin**
- **sp_addremotelogin**
- **sp_dropremotelogin**
- **sp_addserver**
- **sp_dropserver**

Increased Number of Parameters

Adaptive Server 15.7 ESD #1 increases from 2048 to 32767 the maximum number of parameters you can include in dynamic SQL statements and parameterized language statements.

Multiple Listeners on Windows

Adaptive Server versions 15.7 ESD #1 and later start listener tasks on every discrete network connection it can identify on the local computer, ensuring that your clients can connect, regardless of the physical network connection they are using.

Windows Supports Adaptive Server High Availability

Adaptive Server version 15.7 running on the 64-bit Windows operating system supports high availability.

New and Changed Configuration Parameters

Adaptive Server version 15.7, ESD #1 adds the **network polling mode** configuration parmeter, and changes the setting for **number of network tasks**.

Table 14. New Configuration Parameter

Configuration Parameter	Description
network polling mode	When **network polling mode** is set to **threaded**, Adpative Server spawns a separate thread for each network task configured that performs polling. When set to **inline**, one of the engines performs the polling.

Table 15. Changed Configuration Parameter

Configuration Parameter	Description
number of network tasks	**number of network tasks** functions only when **network polling mode** is set to threaded.

Version 15.7

Adaptive Server® version 15.7 introduces many new features and enhancements.

Application Functionality Configuration Group

Adaptive Server version 15.7 adds the Application Functionality configuration group to the configuration file.

These are the Application Functionality configuration parameters

- `enable functionality group`
- `select for update`
- `streamlined dynamic SQL`
- `enable inline default sharing`
- `enable permissive unicode`
- `quoted identifier enhancements`

Use `enable functionality group` to enable or disable all configuration parameters in this group. Enable or disable individual configuration parameters to overwrite the group value.

See "Setting Configuration Parameters" in the *System Administration Guide, Volume 1*.

New Adaptive Server Kernel

Adaptive Server version 15.7 and later includes two kernels: a threaded kernel and a process kernel.

The kernel for which you configure Adaptive Server determines the mode in which Adaptive Server runs:

- Threaded mode – Adaptive Server runs as a single multithreaded operating system process, and processes SQL queries with engines running on threads in thread pools. Threaded mode utilizes threads without engines to manage I/O. Administrators can configure additional thread pools to manage workload.
- Process mode – The legacy kernel on which Adaptive Server previously ran. In process mode, Adaptive Server runs as multiple operating system processes that cooperate to work as a single server. Process mode uses engines to manage I/O, and administrators configure engine groups to manage workload.

For many workloads, threaded mode uses significantly less CPU than process mode, delivering the same—or better—performance. Threaded mode does not require as much task-

to-engine affinity, thereby delivering more consistent performance in a mix of I/O- and CPU-intensive workloads.

The threaded kernel allows Adaptive Server to take advantage of parallel hardware and support systems that have more processors, processor cores, and hardware threads than earlier-version kernels. Although version 15.7 changes the kernel, the query processor remains the same. To run in threaded kernel mode, you need not change most scripts written for earlier versions of Adaptive Server, although few commands and stored procedures have changed. Applications are completely compatible with threaded mode.

Threaded mode is the default mode for Adaptive Server, and the mode that Sybase® recommends. Adaptive Server on the Windows platform runs only in threaded mode.

See the *System Administration Guide: Volume 2* for information about configuring Adaptive Server for threaded mode.

Compressing Data in Adaptive Server

Adaptive Server version 15.7 introduces data compression, which lets you use less storage space for the same amount of data, reduce cache memory consumption, and improve performance because of lower I/O demands.

You can compress large object (LOB) and regular data.

After you create a compressed table or partition, Adaptive Server compresses any subsequently inserted or updated data (that is, existing data is not already compressed). If Adaptive Server cannot efficiently compress the inserted data, the original row is retained. If newly inserted or updated LOB data occupies space that is smaller than or equal to a single data page, Adaptive Server does not compress this data.

You need not uncompress data to run queries against it. You can insert, update, and delete compressed data; running **select** or **readtext** statements on the compressed column returns decompressed rows. Because there is less data for Adaptive Server to search, there are fewer I/Os, improving the efficiency of data storage.

Data compression is a separately licensed option. See the *Compression Users Guide.*

New Security Features

Adaptive Server version 15.7 adds these features for security: end-to-end CIS Kerberos authentication, dual control of encryption keys and unattended startup, securing logins, roles and password management extensions, and login profiles.

End-to-End CIS Kerberos Authentication

Adaptive Server version 15.7 includes end-to-end Kerberos authentication support for remote Adaptive Server connections through the component integration system (CIS).

End-to-end (CIS) Kerberos authentication allows a Kerberos V5 user logged in to Adaptive Server using Kerberos authentication to connect to a remote Adaptive Server using the Kerberos unified login authentication when:

- Requesting an RPC to the Adaptive Server
- Issuing CIS passthrough connections
- Issuing general distributed query processing requests to remote Adaptive Servers using CIS

Adaptive Server supports these optional security services for Kerberos connections to remote ASE servers using CIS:

- Message confidentiality
- Message integrity
- Mutual authentication

See the *Security Administration Guide*.

Dual Control of Encryption Keys and Unattended Startup

Adaptive Server version 15.7 adds the dual control of encryption keys and unattended startup features.

Changes for dual control and split knowledge, and unattended startup include:

- The `master` and `dual master` system keys are database-level keys, created by users with the sso_role or keycustodian_role, and are used as key-encryption keys (KEKs) for user-created encryption keys to achieve better security and split knowledge for data encryption keys. The `master` key replaces the current system encryption password, which Adaptive Server continues to support for backward compatibility purposes. Sybase recommends that users no longer use system encryption passwords to encrypt data encryption keys.
- The ability to supply passwords for the `master` and `dual master` keys with SQL commands and through a private file. Passwords for the master keys are non-persistent: they are not stored in the database.
- The ability to protect all user-created keys through dual control and split knowledge.

See the *Encrypted Columns Users Guide*.

Securing Logins, Roles, and Password Management Extensions

Adaptive Server version 15.7 includes extension for securing logins, roles, and password management.

Adaptive Server improves logins, roles and their passwords through:

- Stronger encryption for role passwords stored on disk.
- Locked roles accounting.
- ISO 8601 duration specification for login, role, and global password policy options.
- Password management extensions such as customizing the Rivest-Shamir-Adleman (RSA) keypair regeneration period, extending password complexity checks to roles passwords, and extending password management support in High Availability environments.
- Auditing enhancements to role definition, activation, role locking due to failed activation attempts, and password management extensions such as RSA keypair regeneration.

See the *Security Administration Guide*.

Login Profiles

Adaptive Server 15.7 adds login profiles, which are SQL-defined containers for login attributes and their values.

Login accounts are defined and governed by attributes. For example, these attributes are associated with login accounts when adding or modifying a login account:

- The database to use
- Which roles to automatically activate
- The language to apply
- The login script to invoke when you log in to Adaptive Server

You can associate some attributes with all login accounts by specifying them in the default login profile, and associate other attributes with a specific set of login accounts by specifying them in a login profile and associating this login profile with login accounts.

See the *Security Administration Guide*.

Employee Lifecycle Management

System security officers or database owners can transfer the ownership of database objects using **alter... modify owner**.

The command provides a way for a database administrator to manage the assignment of objects due to employee changes or to separate the creation ownership of database objects.

External Passwords and Hidden Text

Adaptive Server 15.7 provides strong encryption for external login passwords and hidden text, using the AES-256 symmetric encryption algorithm.

Strong encryption for external passwords applies to passwords for the following:

- Replication Agents – Passwords for replicated databases.
- CIS – Passwords for remote descriptors and logins.
- Job Scheduler – Passwords for Job Scheduler Agent.
- RTMS – Passwords for Real Time Messaging Services.
- Secure Socket Layer (SSL) and Lightweight Directory Access Protocol (LDAP) – Passwords for SSL and LDAP access account. Passwords are administered using stored procedures sp_ldapadmin and sp_ssladmin can be secured.

Abstract Plans in Cached Statements

Adaptive Server version 15.7 introduces the ability to save abstract plan information in the statement cache.

In this example, which includes an abstract plan, the hash table saves **select * from t1 plan '(use optgoal allrows_mix)'**, as shown in the `SQL TEXT` line:

```
1> select * from t1 plan '(use optgoal allrows_mix)'
2> go
1> dbcc prsqlcache
2> go

Start of SSQL Hash Table at 0x0x1474c9050

Memory configured: 1000 2k pages        Memory used: 17 2k pages


Bucket# 243 address 0x0x1474c9f80

SSQL_DESC 0x0x1474cd070
ssql_name *ss0626156152_0290084701ss*
ssql_hashkey 0x0x114a575d       ssql_id 626156152
ssql_suid 1             ssql_uid 1      ssql_dbid 1      ssql_spid 0
ssql_status 0x0xa0      ssql_parallel_deg 1
ssql_isolate 1          ssql_tranmode 32
ssql_keep 0             ssql_usecnt 1   ssql_pgcount 6
SQL TEXT: select * from t1 plan '(use optgoal allrows_mix)'

End of SSQL Hash Table
```

In versions of Adaptive Server earlier than 15.7, the `SQL TEXT` line included only the `select * from t1` command, without the plan clause.

See the *Performance and Tuning Series: Query Processing and Abstract Plans*.

Shrink Log Space

In Adaptive Server version 15.7 and later, **alter database** includes the **log off** parameter, which removes unwanted portions of a database log, allowing you to shrink log space and free storage without re-creating the database.

log off may be particularly helpful after running the fully logged option for database operations, such as **select into**, **alter table**, or **reorg rebuild**, when the database ends up with extra allocated space that is no longer needed.

See "Fully Recoverable DDL" in this document and "Creating and Managing User Databases" in the *System Administration Guide, Volume 2.*

Displaying Currently Set Switches with sysoptions

Adaptive Server version 15.7 adds the `number` column to the `sysoptions` table, which contains the switch ID for currently set switches.

`sysoptions` shows these switches:

- Trace flags set in the runserver file with the **-T** flag
- Trace flags set with `dbcc traceon(`*`flag_number`*`)` or **set switch serverwide on**
- Trace flags and switches set for a specific system process ID (spid) with `set switch on`

`sysoptions` shows only the switches that are visible to the user. That is, users cannot see switches set privately by other spids. The value for number is Null for all option categories other than switches.

Changes for Large Objects

Adaptive Server version 15.7 includes changes for large objects (LOBs), such as storing in-row LOB columns for small text, image, and unitext datatypes, storing declared SQL statements containing LOBs, indirectly referencing a LOB in Transact-SQL statements, and allowing checking for null values of large objects.

In-Row Off-Row LOB

Adaptive Server 15.7 supports the storage of in-row LOB columns for `text`, `image`, and `unitext` datatypes when they are small, and subject to available space in the page.

When a LOB expands in size or its space is used for other in-row columns (such as those used for `varchar` and `varbinary` datatypes), Adaptive Server seamlessly migrates the in-row LOB data to off-row storage, automatically replacing the data with an in-row text pointer.

In Adaptive Server 15.7, you can use:

- **create table** to specify in-row storage of LOB columns
- **alter table** to perform modifications of how LOB columns are stored
- **create database** or **alter database** commands to manage database-wide in-row lengths for LOB columns

See "In-Row, Off-Row LOB" in the *Transact-SQL Users Guide*.

Using Large Object text, unitext, and image Datatypes in Stored Procedures

Once they are declared, Adaptive Server stores SQL statements containing LOBs.

In Adaptive Server version 15.7 and later, you can:

- Declare a large object (LOB) `text`, `image`, or `unitext` datatype for a local variable, and pass that variable as an input parameter to a stored procedure.
- Prepare SQL statements that include LOB parameters.

See the *Reference Manual: Building Blocks*.

Using LOB Locators in Transact-SQL Statements

Large object (LOB) locators let you indirectly reference a LOB in Transact-SQL statements rather than referencing the LOB itself.

Because the size of a `text`, `unitext`, or `image` LOB can be many megabytes, using an LOB locator in Transact-SQL statements reduces network traffic between the client and Adaptive Server, and reduces the amount of memory otherwise needed by the client to process the LOB.

Adaptive Server 15.7 allows client applications to send and receive locators as host variables and parameter markers.

When you create a LOB locator, Adaptive Server caches the LOB value in its memory and generates an LOB locator to reference it.

After a LOB locator is created, it remains valid for the duration of the transaction in which it was created. Adaptive Server invalidates the locator when the transaction commits or is rolled back.

LOB locators use three different datatypes

- `text_locator` – for text LOBs
- `unitext_locator` – for unitext LOBs
- `image_locator` – for image LOBs.

See "Using and Creating Datatypes" in the *Transact-SQL Users Guide*.

Extension to where Clause for Large Objects

The `where` clause is extended to allow checking for null values of large objects.

See the *Reference Manual: Commands.*

Showing Cached Plans in XML

The **show_cached_plan_in_xml** function returns a `showplan` output in XML for a statement in cache.

You must enable the statement cache before using **show_cached_plan_in_xml**.

See "Displaying Query Optimization Strategies and Estimates" in the Performance and Tuning Series: *Query Processing and Abstract Plans*

Padding a Character Field Using str

In Adaptive Server version 15.7 the **decimal** parameter of the **str** function has been extended to allow a field to be padded with a specified character or numeric.

See the *Reference Manual: Building Blocks*.

Changes to select for update

Adaptive Server version 15.7 supports **select for update** to exclusively lock rows for subsequent updates within the same transaction, and for updatable cursors. This prevents other concurrent tasks from updating these rows and from blocking the subsequent update. **select for update** is supported at isolation levels 1, 2, and 3.

You can issue **select for update** as a language statement outside of a cursor context. With both language statements and cursors, you must execute **select for update** within a **begin transaction** command or in chained mode.

If you run **select for update** within a cursor context, the cursor **open** and **fetch** statements must be within the context of a transaction, otherwise, Adaptive Server reverts to pre-15.7 functionality.

See "Queries: Selecting Data from a Table" in the *Transact-SQL Users Guide*.

Creating Nonmaterialized, Non-null Columns

Adaptive Server version 15.7 allows you to create nonmaterialized, non-NULL columns.

Nonmaterialized columns exist virtually, but are not physically stored in the row. Use nonmaterialized columns the same as any other column, selecting, updating, and referring to them in SQL queries, or using them as index keys.

Adaptive Server treats nonmaterialized columns similar to the way it treats null columns: if a column is not physically present in the row, Adaptive Server supplies a default. The default for a nullable column is null, but the default for a nonmaterialized column is a user-defined non-NULL value.

See "Adding, Changing, Transferring, and Deleting Data" in the *Transact-SQL Users Guide*.

Sharing Inline Defaults

Adaptive Server 15.7 supports sharing inline defaults between different tables if the tables are in the same database.

Before creating a new inline default, Adaptive Server looks for an existing shareable inline default having the same value in the database belonging to the same user. If Adaptive Server finds an existing shareable default, it binds this object to the column instead of creating a new default. However, if Adaptive Server does not find an existing shareable inline default, it creates a new default.

Adaptive Server cannot share inline defaults in `tempdb`.

See "Defining Defaults and Rules for Data" in the *Transact-SQL Users Guide*.

Retain Monitoring Data

Adaptive Server version 15.7 does not store the descriptors for some objects in the metadata cache. Instead, it retains monitoring data stored in the descriptors, thus improving query performance.

The metadata cache is a limited resource and can hold a limited number of object descriptors. Adding descriptors to the cache may cause other descriptors to be flushed from the cache.

Adaptive Server discards the descriptor for an object that is not already in the cache, instead of consuming resources when you run these functions:

- `data_pages`
- `used_pages`

- `reserved_pages`
- `object_id`
- `row_count`
- `datachange`
- `derived_stat`

See the *Reference Manual: Building Blocks*.

Analyze Dynamic Parameters

Adaptive Server version 15.7 allows you to analyze dynamic parameters (which are indicated by question marks) before running a query, helping you avoid inefficient query plans.

Analyze the dynamic parameters using:

- *@@lwpid* global variable – returns the object ID of the most recently prepared lightweight procedure that corresponds to a dynamic SQL prepare statement.
- *@@plwpid* global variable – returns the object ID of the next most recently prepared lightweight procedure that corresponds to a dynamic SQL prepare statement.
- **show_dynamic_params_in_xml** – displays information about parameters in dynamic SQL statements.

Using the value provided by *@@plwpid* as the value for the **show_dynamic_params_in_xml** object_id parameter, Adaptive Server displays information about the dynamic parameters in the query. Continue refining the query plan until you determine the parameters that provide you with the best query plan.

See "Displaying Query Optimization Strategies and Estimates" in the *Performance and Tuning Series: Query Processing and Abstract Plans*.

Monitor Lock Timeouts

Adaptive Server version 15.7 allows you to monitor lock timeouts.

Adaptive Server version 15.7 adds this information for tracking locks:

- The `monLockTimeout` monitoring table provides information about timeout lock requests, such as lock types, owners, locks status, and so on. See the *Reference Manual: Tables*
- These parameters configure Adaptive Server to collect lock wait timeout information and make it available for the `monLockTimeout` table:
  - `lock timeout pipe active`
  - `lock timeout pipe max messages`

See "Setting Configuration Parameters" in the *System Administration Guide, Volume 1*.

Truncate Trailing Zeros

Adaptive Server version 15.7 includes the **disable varbinary truncation** configuration parameter, which enables or disables the truncation of trailing zeros from `varbinary` and `binary` null data.

By default, **disable varbinary truncation** is off for the server.

See "Adding, Changing, Transferring, and Deleting Data" in the *Transact-SQL Users Guide*.

Fully Recoverable DDL

Adaptive Server version 15.7 allows you to use **dump transaction** to fully recover the operations that earlier versions of Adaptive Server minimally logged.

These operations are recoverable with **dump transaction**:

- **select into**
- **alter table** commands that require data movement
- **reorg rebuild**

Run **sp_dboption** in the master database to fully log commands that are, by default, minimally logged.

See the *Reference Manual: Commands*.

Transfer Rows from Source to Target Table Using merge

Adaptive Server 15.7 introduces a **merge** command, which allows you to transfer rows from a source table into a target table

See the *Reference Manual: Commands*.

View Statistics and Histograms with sp_showoptstats

sp_showoptstats allows you to extract and display, in an XML document, statistics and histograms for various types of data objects from system tables such as `systabstats` and `sysstatistics`.

See the *Reference Manual: Procedures* and "Statistics Tables and Displaying Statistics with optdiag" in the *Performance and Tuning Series: Improving Performance with Statistical Analysis.*

Changes to Cursors

Adaptive Server version 15.7 includes changes to cursor locks, how cursors manage transactions, and how it declares cursor statements.

Release Cursor Locks at Cursor Close

Adaptive Server 15.7 includes the **declare cursor ... release_locks_on_close** option to release cursor locks at isolation levels 2 and 3 when the cursor is closed, even if the transaction is active.

See the *Reference Manual: Commands* and "Cursors: Accessing Data" in the *Transact-SQL Users Guide*.

Enhanced Transaction Support for Cursors

Adaptive Server 15.7 and later changes the way cursors support transactions:

Adaptive Server:

- Does not automatically close an open cursor declared with a **for update** clause if you commit a transaction. To close a read-only cursor when a transaction is committed, set the **close on end tran** option
- Supports **fetch** operations on open cursors after the transaction has been committed

See the *Reference Manual: Commands* and "Cursors: Accessing Data" in the *Transact-SQL Users Guide*.

Monitor Cursor Statements

Adaptive Server version 15.7 monitors cursor statements based information from the monCachedStatement monitoring table.

For example, although you declare the `new_cursor` cursor (specified by "sq0267364184_1108036110ss"), **sp_cursorinfo** does not display its plan:

```
declare new_cursor cursor for select id from sysroles
go
sp_cursorinfo
go
Cursor name 'new_cursor' is declared on procedure
'*sq0267364184_1108036110ss*'
The cursor is declared as NON-SCROLLABLE cursor.
The cursor id is 983044.
The cursor has been successfully opened 0 times.
The cursor will remain open when a transaction is committed or rolled
back.
The number of rows returned for each FETCH is 1.
The cursor is updatable.
This cursor is declared on a stored procedure. It is presently using
```

```
'860'
bytes. However, the memory usage will increase when the cursor is
opened because
the query plan will be associated with the cursor at that time.
```

Adaptive Server compiles cursors when you open them.

Use the **enable functionality group** configuration parameter to enable and disable monitoring cursor statements.

See "Cursors: Accessing Data" in the *Transact-SQL Users Guide*.

Nested select Statement Enhancements

Adaptive Server 15.7 expands the abilities of the asterisk (*****).

In Adaptive Server 15.7 and later, you can use an asterisk in a nested **select** statement that is not an **exists** subquery as long as the asterisk:

- Is the only item in the **select** statement
- Resolves to a single table column for the nested query

In addition, you can:

- Restrict the selected columns in your nested query to only those belonging to a specific table by using the *qualifier*.***** format, where *qualifier* is one of the tables in the `from` clause.
- Use the asterisk in a nested query that includes a group by clause.

When an asterisk resolves to a single table column for the nested query, the query is equivalent to explicitly using a single table column.

See "Queries: Selecting Data from a Table" in the *Transact-SQL Users Guide*.

Changes to Commands and System Procedures in Chained Transaction

Adaptive Server versions 15.7 allows some system procedures to run in sessions that use chained transaction mode.

- These system procedures can run in sessions using chained transaction mode if there are no open transactions:
  - **sp_configure**
  - **sp_engine**
  - **sp_rename**
- These system procedures can run in sessions using chained transactions after you use **sp_procxmode** to change the transaction mode to `anymode`:

  - **sp_addengine**
  - **sp_dropengine**
  - **sp_showplan**
  - **sp_sjobcontrol**
  - **sp_sjobcmd**
  - **sp_sjobcreate**
- **sp_sjobdrop** can run in sessions using chained transaction mode, but fails if you execute it during an open transaction.

When you execute these stored procedures, Adaptive Server implicitly commits the changes performed by these stored procedures when there are no open transactions, so you need not issue a **commit** or **rollback.**

If an open transaction exists when you issue:

- **sp_rename**, **sp_configure**, **sp_engine**, **sp_addengine**, or **sp_dropengine** – the procedures fail with error 17260 because they cannot run within a transaction.
- **sp_sjobcontrol**, **sp_sjobcmd**, **sp_sjobcreate**, **sp_sjobdrop**, or **sp_showplan** – Adaptive Server leaves the transaction open after the procedure executes. You must explicitly issue `commit` or `rollback` for the entire transaction. If these procedures receive an error when they execute, they roll back only the operations performed inside the procedure, but do not roll back the operations performed before they execute, even though the operations are performed in the same transaction.

Use **set chained {on | off}** to set the chained mode for the session.

See the *Reference Manual: Commands* and the *Reference Manual: Procedures*.

Expanded Variable-Length Rows

Adaptive Server version 15.7 redefines data-only locked (DOL) columns to use a row offset of up to 32767 bytes. You must configure Adaptive Server for a logical page size of 16K to create wide, variable-length DOL rows.

By default, Adaptive Server does not use wide, variable-length DOL rows. Enable wide, variable-length DOL rows for each database using:

```
sp_dboption database_name, 'allow wide dol rows', true
```

See "Data Storage" in the *Performance and Tuning Series: Physical Database Tuning*.

Changes to like Pattern Matching

Adaptive Server version 15.7 allows you to treat square brackets individually in the `like` pattern-matching algorithm.

For example, matching a row with `'[XX]'` in earlier versions of Adaptive Server required you to use:

```
select * from t1 where f1 like '[[]XX[]]
```

However, in Adaptive Server 15.7, you can also use:

```
select * from t1 where f1 like '[[]XX]'
```

Changes to Quoted Identifiers

In Adaptive Server 15.7 and later, you can use quoted identifiers for tables, views, column names, index names, and system procedure parameters.

In versions earlier than 15.7, Adaptive Server treated “ident” as an identifier that used nonalphanumeric characters delimited with double quotes (quoted identifiers) or square brackets. These identifiers could be used only for table, view, and column names.

See the *Reference Manual: Blocks*.

Allowing Unicode Noncharacters

In Adaptive Server version 15.7, the **enable permissive unicode** configuration parameter, which is a member of **enable functionality group**, allows you to ignore Unicode noncharacters.

When you enable this feature, Unicode noncharacters are not detected in:

- Parameters
  - Presented as `univarchar` and `unitext` (UTF-16) datatypes
  - Presented as `varchar` and `text` (UTF-8) datatypes
  - As parameters to dynamic SQL statements
  - As input to parameterized language statements
  - As input to parameterized language statements
- String literals when the server’s character set is UTF-8
- Escaped string literals (those prefixed with U&), regardless of the server’s character set
- Conversion processes between `unichar` (UTF-16) and `varchar` (UTF-8) in either direction

In addition, Unicode noncharacters are acceptable in simple expressions such as comparisons, where they sort higher than legal Unicode characters.

In versions of Adaptive Server earlier than 15.7, the `unichar`, `univarchar`, `unitext`, `char`, `varchar`, and `text` datatypes under the utf-8 default character set did not accept Unicode noncharacters (code points permanently reserved for internal use).

See "Setting Configuration Parameters" and "Configuring Client/Server Character Set Conversions" in the *System Administration Guide, Volume 1*.

Reduce Query Processing Latency

The query processing layer in Adaptive Server 15.7 enables multiple client connections to reuse or share dynamic SQL lightweight procedures (LWPs).

Adaptive Server uses the statement cache to store dynamic SQL statements converted to LWPs. Because the statement cache is shared among all connections, dynamic SQL statements can be reused across connections. These statements are not cached:

- **select into** statements.
- insert-values statements with all literal values and no parameters.
- Queries that do not reference any tables
- Individual prepared statements that contain multiple SQL statements. For example

```
statement.prepare('insert t1 values (1) insert
t2 values (3)');
```

- Statements that cause instead-of triggers to fire

Use the **streamlined dynamic SQL** or **enable functionality group** configuration parameters to enable this feature.

See "Memory Use and Performance" in the *Performance and Tuning Series: Basics* and "Setting Configuration Parameters" in the *System Administration Guide, Volume 1*.

The sybdiag Utility

Adaptive Server 15.7 adds the **sybdiag** utility, a Java-based tool that collects comprehensive Adaptive Server configuration and environment data. Sybase Technical Support uses this information to diagnose server issues, thus expediting customer cases.

sybdiag connects to an Adaptive Server and executes system procedures such as **sp_configure** and queries to tables like `monLicense`. It collects operating system and platform diagnostic information by executing commands such as **ps**, **vmstat**, and **netstat**.

sybdiag generates a `.zip` output file comprising HTML and data files that can be unzipped and viewed in a Web browser. The information collected includes operating system and

environment data, Adaptive Server configuration and monitoring data, and Adaptive Server files and scripts.

sybdiag does not collect Adaptive Server or operating system data for logins, passwords, or user lists, and does not collect information from application database tables.

See the *Utility Guide.*

The Optimizer Diagnostic Utility

Adaptive Server version 15.7 includes the **sp_opt_querystats** system procedure, which allows you to analyze the query plan generated by the Adaptive Server optimizer and the factors that influenced its choice of a query plan.

This analysis helps determine if elements in the query or the execution environment affect how Adaptive Server executes the query and its performance. You need not run the selected query to perform the analysis.

sp_opt_querystats provides this information:

- The query plan generated by showplan
- Enabled traceflags and switches
- I/O activity for the query generated by set statistics io
- Missing statistics found for any of the tables involved in the query
- The estimated plan cost calculated by the optimizer
- The final plan and cost estimations calculated by the optimizer
- The abstract plan for the query
- The result of the query if the result set is executed (for example, if `noexec` is not on)
- The logical operator tree for the query generated by `set` option `show`
- Query execution time generated by **set statistics time**
- After you execute the query, the query execution time generated by `set statistics time`

You must install and configure the Job Schedule to run **sp_opt_querystats.**

See "Controlling Optimization" in the *Performance and Tuning Series: Query Processing and Abstract Plans*.

System Changes for Adaptive Server Version 15.7

Adaptive Server 15.7 includes changes to commands, functions, system procedures, configuration parameters, system tables, monitoring tables, and global variables.

Commands

Adaptive Server 15.7 contains new and changed commands.

Table 16. New commands

Command	Description
alter login	Changes the attributes of a login account
alter login profile	Changes the attributes of a login profile
alter...modify owner	Transfers the ownership of database objects from one owner to another
alter thread pool	Alters a thread pool
create login	Creates a login account; specifies a password, a login profile for the account, and user-supplied parameters to be assigned to the account
create login profile	Creates a login profile with specified attributes
create thread pool	Creates a user-defined thread pool
deallocate locator	Deletes a large object (LOB) stored in memory and invalidates its LOB locator
drop login	Drops a login account or list of accounts
drop login profile	Drops a login profile or list of login profiles
drop thread pool	Drops a user-defined pool
merge	Transfers rows from a source table into a target table
select for update	Exclusively locks rows for subsequent update within the same transaction
truncate lob	Truncates a LOB to a specified length

Table 17. Changed commands

Command	Change
alter database changes	• allows you to change the compression setting at the database level • **alter database .. inrow_LOB_length** – allows you to change the length of in-row LOB columns database-wide • **alter database ... log off** – removes unwanted portions of a database log, allowing you to shrink log space and free storage without re-creating the database

Command	Change
alter encryption key	• **master** and **dual master** –indicate you are alteing a master or dual master encryption key • **master key** – indicates you are altering the encryption key with the master key • **[no] dual_control** – indicates whether the new key is encrypted using dual control. • **for recovery** – indicates the key copy will be used to recover the master key in case of a lost password • **for automatic_startup** – indicates the key copy will be used to access the master or dual master key after the server starts • **regenerate key** – replaces the raw key value for the master or dual master keys with a new raw key, and re-encrypts all column encryption keys encrypted by the master or dual master keys
alter table	• allows you to change the compression attributes for tables, columns, and partitions. • **alter table ... not materialized** – indicates you are creating a non-materialized column • **alter table ... add lob-colname** – allows you to define newly added nullable LOB columns as in-row, and specify its length • **alter table ... modify lob-colname** – allows you to modify an existing LOB column from off-row to in-row • **alter table ... modify off row | in row** – specifies whether the Java-SQL column is stored separately from the row, or in storage allocated directly in the row
Concatenation operators	The **+** and **||** Transact-SQL operators accept LOB locators as expressions for a concatenation operation. The result of a concatenation operation involving one or more locators is a new LOB locator with the same datatype as that referenced by the input locator.
create database	• **compression =** indicates the level of compression to be applied to newly created tables or partitions. • **lob_compression =** *value* – Determines the compression level for the newly created table. Selecting off means the table does not use LOB compression. • **inrow_lob_length =** *value* – specifies the number of bytes. The range of valid values for inrow_lob_length is 0 through the logical page size of the database.

Command	Change
create encryption key	• **master** and **dual master** – indicate you are creating a master or dual master encryption key • **passwd system_encr_passwd | master key** – indicates you are using system encryption password or the master key for the password • **[no] dual_control** – indicates whether the new key is encrypted using dual control.
create table ... [in row [(length)] | off row]	allows you to create a compressed table **create table** lets you specify that the data in a LOB column be kept in the row, instead of stored off-row.
declare cursor ... [release_locks_on_close]]	Allows you to configure the lock-releasing behavior of each cursor so that the shared locks can be released when the cursor is closed, even if the transaction is active.
drop encryption key	**[dual] master** – indicates you are dropping a master or dual master key
dump database ... with shrink_log	Allows you to remove any holes at the end of a database, regardless of whether the database is in a dump sequence.
like clause in a **where** clause	**where** clause accepts text and unitext LOB locators, but not image LOB locators, for the *variables* expression and *match_string.*
select into ... [in row [(length)] | off row]	Sets or changes the in-row chracteristics for the text columns in the target table. If you do not specify length, Adaptive Server uses the configured default in-row length.
set	**set** adds • **send_locator [on | off]** – specifies whether Adaptive Server sends the LOB or the locator that references the LOB in a result set sent to the client. • **cis_rpc_handling {on | off}** – makes CIS the remote procedure call (RPC) handling mechanism the default mechanism for Shared Disk Cluster (SDC) handling • **encryption passwd <char_literal> for key [dual] master** – sets the password for the master or dual master key
where clause extension to support LOBs	**where** clauses in select, insert, update, and delete statements can include a condition for null large objects (LOBs).

See the *Reference Manual: Commands*.

Functions

Adaptive Server 15.7 contains new and changed functions.

Table 18. New functions

Function	Description
dol_downgrade_check	Returns the number of data-only-locked (DOL) tables in the specified database that contain variable-length columns wider then 8191 bytes
create_locator	Explicitly creates a locator for a specified large object (LOB)
locator_literal	Identifies a binary value as a locator literal
locator_valid	Determines whether a LOB locator is valid
lprofile_id	Returns the login profile ID for the specified login profile name, or the login profile ID for the login profile associated with the current login or specified login name
lprofile_name	Returns the login profile name for the specified login profile ID, or the login profile name for the login profile associated with the current login or specified login suid
return_lob	Dereferences a locator, and returns the LOB referenced by that locator
setdata	Overwrites some or all of a LOB
show_cached_plan_in_xml	Displays, in XML, the executing query plan for queries in the statement cache
show_dynamic_params_in_xml	Returns the text of a query in XML format

Table 19. Changed functions

Function	Description
charindex	charindex adds support for the `text_locator`, `unitext_locator`, and `image_locator` LOB locator datatypes and the **start** option.
charlength	**charlength** supports the text_locator and unitext_locator datatypes.
datalength	**datalength** accepts the `text_locator`, `unitext_locator`, and `image_locator` datatypes.

Function	Description
patindex	**patindex** accepts the `text_locator` and `unitext_locator` datatypes.
show_cached_plan_in_xml	**show_cached_plan_in_xml** expands the scope of the **statement_id** parameter to accept object IDs that refer to any lightweight procedure, not only those in the statement cache.
str	The decimal parameter of the **str** function has been expanded to support padding of the output with a character or numeric to the specified length.
textptr	Because the in-row/off-row LOB feature can split or shrink the data page of an allpages-locked table with a clustered index, causing data rows—incuding in-row LOB columns that reside in those data rows—to move to different pages, the **textptr** text pointer value of such in-row LOB columns before the split or shrink operation differs from that of the same column after such an operation. For Adaptive Server 15.7 and later, the **textptr** value returned for an in-row LOB column residing in a data-only-locking data row that is row-forwarded remains unchanged and remains valid after the forwarding.
textvalid	You can use **textvalid** on the returned text pointer for both in-row and off-row LOBs, returning 1 if the text pointer points to a valid LOB column, and 0 if the LOB column is invalid.

See the *Reference Manual: Building Blocks*.

System Stored Procedures

Adaptive Server 15.7 contains new and changed system procedures.

Table 20. New system stored procedures

System stored procedures	Description
sp_merge_dup_inline_default	Removes existing duplicate inline default objects, converting the unique inline defaults to sharable inline default objects
sp_opt_querystats	Returns a performance analysis for the selected query
sp_securityprofile	Lists the attributes or bindings associated with a login profile

System stored procedures	Description
sp_showoptstats	Extracts and displays statistics and histograms for various data objects from system tables such as systabstats and sysstatistics

Table 21. Changed system stored procedures

System stored procedures	Description
sp_dboption	• **enforce dump tran sequence** – prevents operations that disallow a subsequent dump transaction • **allow wide rows** –configures databases to allow wide, variable-length data-only locked (DOL) rows • **full logging for all** – fully log commands that are minimally logged by default (**select into**, **alter table**, and **reorg rebuild**)
sp_displaylogin	Displays • The login profile name associated with a login account. • The name of the default login profile if there is no login profile directly associated with the login account but there is a default login profile The login overrides the **sp_addlogin** and **sp_modifylogin default database, default language, authenticate with** and **login script** parameters. If login profiles are ignored, or there is no login profile associated to the login account either directly or through a default login profile, sp_displaylogin displays information in the format of versions earlier than 15.7.

System stored procedures	Description
sp_displayroles	• When run against the current login, **sp_displayroles** displays the roles granted to the login profile to which it is associated. **sp_displayroles** requires the sso_role to view the roles associated with other login profiles. • Displays the roles granted to logins through an associated login profile. A `Grantee` column in the output indicates the login profile name, as applicable. **sp_displayroles** displays the `Grantee` column only if the login has an associated login profile with roles granted to it. • Displays the date when the role was locked, the reason for the lock, and the login ID that locked the role. For password protected roles, **sp_displayroles** displays the role password encryption version.
sp_encryption	• When run by the SSO, key custodian, or the DBO, reports that a key is protected by dual control • **sp_encryption helpkey, master** and **sp_encryption helpkey, 'dual master'** report information about the master and dual master keys, including the existence of a copy encrypted for automatic startup and the existence of a recovery copy • **mkey_startup_file [, {<new_path> | default_location | null} [, {sync_with_mem | sync_with_qrm}]** – displays or sets the master key startup file name and path • downgrade_kek_size [, {“true” | “false”}] – displays or sets **downgrade_kek_size** configuration for the server
sp_help	Displays compression settings at column, table, and partition level. Displays the in-row LOB settings at the column and table level.
sp_helpconstraint	Updated to display information about shareable inline defaults

System stored procedures	Description
sp_helprotect	• Accepts 'master key' and 'dual master key' as object names • Accepts 'Set Encryption Passwd' as a valid permission name • Displays dual and master key permissions
sp_helpuser	*display_object* lists all objects and user-defined datatypes owned by *name_in_db* in the current database
sp_locklogin	Exempted login accounts are no longer locked because of inactivity.
sp_passwordpolicy	• **keypair regeneration period** – specifies the date and time to start the first keypair generation and subsequent frequency of keypair regeneration • **keypair error retry wait/count** – specifies the various configurations you can set for regenerating a key pair after a failed attempt
sp_serveroption	Changes the definition for these options: • use message confidentiality – Sets message confidentiality for all connections to the remote server using Kerberos authentication • use message integrity –Sets message integrity for all connections to the remote server using Kerberos authentication

See the *Reference Manual: Procedures*.

Configuration Parameters

Adaptive Server 15.7 introduces new configuration parameters.

Configuration parameter	Description
automatic master key access	Determines Adaptive Server operates in unattended startup mode
capture compression statistics	Enables the monTableCompression monitoring table to begin capturing compression statistics

Configuration parameter	Description
column default cache size	Determines the size of the cache that Adaptive Server must keep in memory to provide defaults for nonmaterialized columns
compression info pool size	Determines the size of the memory pool used for compression
disable varbinary truncation	Controls whether Adaptive Server includes trailing zeros at the end of varbinary or binary null data
enable console logging	Once enabled, Adaptive Server sends messages to the console separately from the error log after startup
enable functionality group	Enables or disables these features in Adaptive Server versions 15.7 and later: • Shareable inline defaults • Select for update • Quoted identifiers • Unicode noncharacters • Monitor cursor statements • Reduce query processing latency
enable hp posix async i/o	Enables asynchronous I/O on HP-UX 11.31 and later
kernel mode	Determines the mode the Adaptive Server kernel uses, threaded or process
kernel resource memory	Determines the size, in 2K pages, of the kernel resource memory pool from which all thread pools and other kernel resources are allocated memory
lock timeout pipe active	Controls whether Adaptive Server collects lock timeout messages
lock timeout pipe max messages	Determines the number of lock timeout messages Adaptive Server stores, and the amount of memory it allocates for the task
number of disk tasks	Controls the number of tasks dedicated to polling and completing disk I/Os

Configuration parameter	Description
number of network tasks	Controls the number of tasks dedicated to polling and completing network I/Os

See the *System Administration Guide: Volume 1*

System Tables

Adaptive Server 15.7 contains new and changed system tables.

Table 22. Changed system tables

System table	Column added	Description
`sysattributes`	• `object_cinfo2` • `object_date-time`	• Provides a character description for the object • Provides the date and time for the object • *sysattributes* adds the **SP** *object_type*, which stores options related to RSA Key pair regeneration and LR object_type, which stores options related to login profiles
`sysdatabases`		Adds status bits to the `status4` column to indicate database-wide compression settings
`sysoptions`	`number`	Lists the switch ID as an integer
`sysobjects`	`lobcomp_lvl`	• Adds status bits to the `status3` column to indicate database-wide compression settings • `lobcomp_lvl` – compression level of the columns defined for large objects.

System table	Column added	Description
syscolumns	inrowlen	• Adds status bits to the status2 column to indicate if a column is explicitly defined as compressed • inrowlen – a nullable column that stores the user-specified, or derived in-row length for LOB columns created as in-row • lobcomp_lvl – compression level of the columns defined for large objects.
syslogins	crsuid	Server user ID of the creator of login or login profile
syslogins	lpid	Login profile ID
syssrvroles	• lockdate • lockreason • locksuid	• Date and time a role was locked • Reason a role was locked • ID of the user who locked the role
sysservers	srvprincipal	Specifies the remote server Kerberos principal name
syscomments	• syb_syscomm-key_dddddd	

Adaptive Server version 15.7 adds these system tables, which are views of the master database and provide information about the configuration of data caches and pools.

System table	Description
syscacheinfo	Provides information about data caches.
syspoolinfo	Provides information about cache pools.
syscachepoolinfo	Provides a row for each data cache pool that includes configuration information for the data cache. This view is a join between the *syscacheinfo* and *syspoolinfo* views.

See the *Reference Manual: Tables*

Utilities

Adaptive Server 15.7 contains new and changed utilities.

New utilities

Command	Description
sybdiag	**sybdiag** is a Java-based tool that collects comprehensive Adaptive Server configuration and environment data. Sybase Technical Support uses this information to diagnose server issues, thus expediting customer cases.

Changed utilities

Command	Description
sybperf	The 15.7 version of **sybperf** exposes a set of Adaptive Server counters that are more useful for monitoring Adaptive Server performance.

Monitoring Table Changes

Adaptive Server version 15.7 contains new and changed monitoring tables.

New Monitoring Tables

Adaptive Server version 15.7 includes new monitoring tables.

Command	Description
`monDeviceSpaceUsage`	Provides information about the file systems on which database devices are allocated. Space information is available only for file system devices. File system size and free space values are NULL for database devices allocated on raw devices.
`monLockTimeout`	Provides information about lock timeout requests

See the *Reference Manual: Tables*.

Changes to Monitoring Tables

Adaptive Server version 15.7 includes changes to some monitoring tables.

Changes to monCachePool

Monitoring table	Description
`LogicalReads`	Number of buffers read from the pool

Monitoring table	Description
PhysicalWrites	Number of write operations performed for data in this pool (one write operation may include multiple pages)
APFReads	Number of APF read operations that loaded pages into this pool
APFPercentage	The configured asynchronous prefetch limit for this pool
WashSize	The wash size, in kilobytes, for a memory pool

Changes to monCachedProcedures

Monitoring table	Description
ExecutionCount	Number of times Adaptive Server executed the stored procedure plan or tree since it was cached
CPUTime	Total number of milliseconds of CPU time used
ExecutionTime	Total amount of elapsed time (in milliseconds) Adaptive Server spent executing the stored procedure plan or tree
PhysicalReads	Number of physical reads performed
LogicalReads	Number of pages read
PhysicalWrites	Number of physical writes performed
PagesWritten	Number of pages written

Changes to monCachedStatement

Monitoring table	Description
OptimizationGoal	Optimization goal stored in the statement cache
OptimizerLevel	Optimizer level stored in the statement cache

Changes to monDataCache

Monitoring table	Description
Status	Status of the cache

Monitoring table	Description
Type	Type of cache
CacheSize	Total size of cache, in kilobytes
ReplacementStrategy	Cache replacement strategy
APFReads	Number of asynchronous prefetch (APF) reads for this data cache
Overhead	Cache overhead

Columns added to monDeadLock

Column	Description
HeldClientApplName	Value for the *clientapplname* property set by the application holding the lock
HeldClientName	Value of the *clientname* property set by the application holding the lock
HeldClientHostName	Value for the *clienthostname* property set by the application holding the lock
HeldHostName	Name of the host machine on which the application that executed the query holding the lock is running
HeldNumLocks	Number of locks currently held by holding spid
HeldProcDBName	Name of the database in which the stored procedure was executing the blocking process at the time the deadlock occurred, if applicable
HeldProcedureName	Name of the stored procedure the blocking process was executing at the time the deadlock occurred, if applicable
HeldProgramName	Name of program running the process that holds the lock
HeldStmtNumber	Statement number in the SQL batch of the SQL statement holding the lock
ObjectDBName	Name of the database
ObjectID	Unique identifier for the object

Column	Description
WaitApplName	Name of the application waiting for the lock
WaitBatchID	Identifier of the SQL batch executed by the process waiting for the lock when the lock timeout occurred
WaitClientApplName	Value of the *clientapplname* property set by the application waiting for the lock
WaitClientHostName	Value of the *clienthostname* property set by the application waiting for the lock
WaitClientName	Value of the *clientname* property set by the application waiting for the lock
WaitCommand	Category of process or command that the process was executing when it was blocked and then timed out
WaitContextID	Unique context identifier for the process waiting for the lock when it was blocked by another process
WaitHostName	Name of the host running the process waiting for the lock
WaitLineNumber	Line number of the SQL statement in the SQL batch or stored procedure waiting for the lock
WaitProcDBID	Unique identifier for the database in which the stored procedure waiting for the lock resides, if applicable
WaitProcDBName	Name for the database where the stored procedure that is waiting for the lock resides, if applicable
WaitProcedureID	ID of the stored procedure waiting for the lock, if applicable
WaitProcedureName	Name for the stored procedure waiting for the lock, if applicable
WaitProgramName	Name of the program running the process
WaitStmtNumber	Line number in SQL batch waiting for the lock

Column	Description
WaitTranName	Name of the transaction in which the lock was requested

Changes to monErrorLog

Adaptive Server version 15.7 and later change the value for stack traces in the monErrorLog.Severity column. Earlier versions used 0 as a value for stacktraces. In Adaptive Server version 15.7, all rows representing stack traces have a Severity value of 99.

Changes to monLockTimeout

Monitoring table	Description
HeldProgramName	Removed from monLockTimeout
WaitProgramName	Removed from monLockTimeout
HeldProcedureID	Unique object identifier for the stored procedure that the blocking process was executing when the timeout occurred
WaitProcedureID	Unique object identifier for the stored procedure that is waiting for the lock, if applicable

Changes to monOpenObjectActivity

Monitoring table	Description
SharedLockWaitTime	The total amount of time, in milliseconds, that all tasks spent waiting for a shared lock
ExclusiveLockWaitTime	The total amount of time, in milliseconds, that all tasks spent waiting for an exclusive lock
UpdateLockWaitTime	The total amount of time, in milliseconds, that all tasks spent waiting for an update lock
ObjectCacheDate	Indicates the date and time when the object was added to the cache

Changes to monOpenPartitionActivity

Monitoring table	Description
ObjectCacheDate	Indicates the date and time when the object was added to the cache

Changes to monProcess

Monitoring table	Description
`HostName`	Name of the host machine on which the application that started the process is running
`ClientName`	Value of the *clientname* property set by the application
`ClientHostName`	Value of the *clienthostname* property set by the application
`ClientApplName`	Value of the *clientapplname* property set by the application

Changes to monProcessActivity

Monitoring table	Description
`HostName`	Name of the host machine on which the application that executed the query is running
`Application`	Name of the application
`ClientName`	Value of the *clientname* property set by the application
`ClientHostName`	Value of the *clienthostname* property set by the application
`ClientApplName`	Value of the *clientapplname* property set by the application

Changes to monProcessLookup

Monitoring table	Description
`ClientName`	`Value of the clientname property set by the application`
`ClientHostName`	`Value of the clienthostname property set by the application`
`ClientApplName`	Value of the *clientapplname* property set by the application

Changes to monProcessProcedures

Monitoring table	Description
ExecutionCount	Number of times Adaptive Server executed this instance of the stored procedure held in the procedure cache
CPUTime	The amount of CPU time, in milliseconds, that Adaptive Server spent executing the instance of this stored procedure held in the procedure cache
ExecutionTime	Total amount of time, in milliseconds, Adaptive Server spent executing the instance of this stored procedure held in the procedure cache
PhysicalReads	Number of physical reads performed by the instance of this stored procedure held in the procedure cache
LogicalReads	Number of logical reads performed by the instance of this stored procedure held in the procedure cache
PhysicalWrites	Number of physical writes performed by the instance of this stored procedure held in the procedure cache
PagesWritten	Number of pages read by the instance of this stored procedure held in the procedure cache

Changes to monTableColumns

monTableColumns includes these changes:

- Adaptive Server versions 15.7 and later include the column's unit of measurement in the Description column of monTableColumns.
- monTableColumn adds the Label column (datatype varchar(50)), which contains a brief description of the data presented in the column. You can use these values in application user interfaces instead of the actual column names.

Changes to monTables, monTableColumns, monWaitEventInfo, and monWaitClassInfo

These monitoring tables add the Language column (datatype varchar(30)), which allows you to specify the language in which Adaptive Server returns the values of the Description column and the monTableColumns.Label column.

See the *Reference Manual: Tables*.

Global Variables

Adaptive Server version 15.7 includes new global variables.

Command	Description
`@@plwpid`	Returns the object ID of the most recently prepared lightweight procedure
`@@lwpid`	Returns the object ID of the next most recently run lightweight procedure

See the *Reference Manual: Bluilding Blocks*

Version 15.5 Cluster Edition

Adaptive Server® 15.5 Cluster Edition introduces multiple simultaneous failover, distributed transaction management, the **mount** and **unmount** commands, and the ability to use **alter database** to add space to an archived database.

Note: The Cluster Edition does not currently support in-memory databases, relaxed-durability databases, template databases, or minimally-logged DML.

Adaptive Server 15.5 Cluster Edition Feature and Platform Matrix

The feature and platform matrix shows feature availability for supported operating systems in Adaptive Server 15.5 Cluster Edition. A “Y” indicates the feature is supported for that platform.

Adaptive Server Cluster Edition options	HP-UX Itanium 64-bit	IBM AIX 64-bit	Linux Opteron 64-bit	Solaris 64-bit
Security and directory services	Y	Y	Y	Y
Cybersafe Kerberos				Y
Pluggable Authentication Module	Y	Y	Y	Y
Fine-grained access control	Y	Y	Y	Y
LDAP server directory	Y	Y	Y	Y
LDAP user authentication	Y	Y	Y	Y
Platform Native Kerberos				Y
Secure Sockets Layer	Y	Y	Y	Y
MIT Kerberos	Y	Y	Y	Y

Adaptive Server Cluster Edition options	HP-UX Itanium 64-bit	IBM AIX 64-bit	Linux Opteron 64-bit	Solaris 64-bit
Encrypted columns, including fine-grained access control (FGAC)	Y	Y	Y	Y
High availability				
Partitions	Y	Y	Y	Y
In-memory databases				
Tivoli Storage Manager for Backup Server	Y	Y	Y	Y
Active messaging	Y	Y	Y	Y
Enhanced Full-Text Search (EFTS)				
Features included in base Adaptive Server				
Cross-platform dump and load	Y	Y	Y	Y
Job Scheduler	Y	Y	Y	Y
Native XML	Y	Y	Y	Y
IPv6	Y	Y	Y	Y
Java option	Y	Y	Y	Y
Web Services	Y	Y	Y	Y
Distributed transaction management	Y	Y	Y	Y
Content management (external file support)	Y	Y	Y	Y
Archived database access	Y	Y	Y	Y

Multiple simultaneous failover

Adaptive Server Cluster Edition version 15.5 and later support multiple simultaneous instance failures.

Multiple simultaneous failure support occurs when more than one instance fails within a single cluster view, but the cluster remains online and provides the same failover recovery as it does when a single instance fails.

The number of failures cannot be greater than the value for **cluster redundancy level**, a configuration parameter that allows a database administrator to set the maximum number of recoverable simultaneous instance failures for the cluster.

Adding space to an archive database

Adaptive Server Cluster Edition version 15.5 and later supports archive databases. In general, access to an archive database is the same in both a clustered and a nonclustered Adaptive Server. In either scenario, when an archive database runs out of space, use the **alter database** command to add space to the archive database.

In a clustered Adaptive Server, run **alter database** from the same node that is updating the archive database. If you run **alter database** from a different node, Adaptive Server prints an error message with the number of the node that is actually updating the archive database.

Distributed transaction management in the shared-disk cluster

In version 15.5 and later, Adaptive Server supports distributed transaction management (DTM) on its clustered architecture.

The clustered Adaptive Server:

- Is fully compliant with the X/Open XA protocol when it acts as the resource manager (RM), without additional services, such as XA-Server.
- Ensures consistent commit or rollback for transactions that update Adaptive Server data via remote procedure calls (RPCs) and Component Integration Services (CIS).
- Can be part of distributed transactions coordinated by other Adaptive Server installations using the Adaptive Server Transaction Coordination (ASTC) mechanism.
- Can coordinate the distributed transactions across multiple Adaptive Server installations using the ASTC mechanism.

Note: The Cluster Edition does not support the Microsoft Distributed Transaction Coordinator (MSDTC) proprietary protocol.

In general, the user interface for distributed transactions is the same in the Adaptive Server clustered environment as in the nonclustered environment. Applications using DTM on a nonclustered Adaptive Server can use the same applications on the clustered Adaptive Server. See *Using Adaptive Server Distributed Transaction Management Features*.

Although the user interface for distributed transactions on the Cluster Edition is the same as that for nonclustered Adaptive Server, support for distributed transactions on the cluster must take into account the cluster-specific issues described in the *Clusters Users Guide*. For example:

- Using the cluster as the resource manager (RM)
- Requests to nonowner instances
- Handling instance failures
- Using transaction coordination with ASTC
- Impact of connection migration
- System configuration

System Changes for Adaptive Server Version 15.5 Cluster Edition

Adaptive Server 15.5 Cluster Edition supports changes to commands, monitoring tables and configuration parameters.

Changed commands

In Adaptive Server Cluster Edition version 15.5 and later, you can use **mount database** and **unmount database** in a shared-disk cluster.

If an instance fails while **mount database** or **unmount database** is in progress, the command may abort. In this case, you must reissue **mount database** or **unmount database** when the instance failover recovery is complete.

Monitoring Tables

Monitoring tables added for Adaptive Server version 15.5.

Adaptive Server Cluster Edition version 15.5 adds these monitoring tables:

- monTableTransfer – provides historical transfer information for tables in Adaptive Server active memory.
- monInmemoryStorage – used for in-memory databases. For internal purposes only.

The Cluster Edition version 15.0.1 and later include monitoring tables to collect table statistics. These are the monitoring tables included with version 15.5:

- monCIPC – provides summary figures for total messaging within the cluster, as viewed from the current instance or all instances.
- monCIPCEndpoints – provides a detailed summary, giving traffic data for each subsystem within the cluster instance.
- monCIPCLinks – monitors the state of the links between instances in the cluster.
- monCIPCMesh – gives summary figures for the mesh of connections, from the current instance to all other instances in the cluster, on a per-instance basis.
- monCLMObjectActivity – collects cluster lock information.
- monClusterCacheManager – stores diagnostic information about the cluster cache manager daemon running on each instance. monClusterCacheManager reports cluster-wide information on a per-instance basis.
- monCMSFailover – tracks the time at which the cluster membership service (CMS) detects the failure, gets a new cluster view, resynchronizes the heartbeat, posts the failure event, and completes the failure event. There is a row for each instance.
- monDBRecovery – contains rows from all instances in the cluster and contains rows for every database that contributes to recovery.
- monDBRecoveryLRTypes – tracks log records seen during recovery. Contains a row for each log record type for which at least one log record was seen by recovery.
- monFailoverRecovery – contains aggregated failover recovery diagnostic information for the cluster lock manager (CLM), database recovery, and CMS modules.
- monLogicalCluster – displays information about the logical clusters currently configured on the system.
- monLogicalClusterAction – shows all administrative actions against local clusters from start-up until these actions are released.
- monLogicalClusterInstance – displays information about the many-to-many relationship between instances and logical clusters.
- monLogicalClusterRoute – displays information about the configured routes (application, login, and alias bindings). You need not have the mon_role role to query this monitor table.
- monPCM – tracks the peer coordination module (PCM) client activities in the cluster (for example, the number of fragment that were sent and received), and contains a row for each PCM client.
- monProcessMigration – displays information about the connection currently migrating.
- monSysLoad – provides trended statistics on a per-engine basis. You need not have the mon_role role to query this monitor table.
- monTempdbActivity – provides statistics for all open local temporary databases, including global system tempdb when the instance is started in tempdb configuration mode.
- monWorkload – displays the workload score for each logical cluster on each instance according to its load profile.

- monWorkloadPreview – provides an estimate of how a load profile impacts the workload score without enabling the profile. monWorkload includes one row for each logical cluster and instance on which this logical cluster is running.
- monWorkloadProfile – displays currently configured workload profiles. You need not have the mon_role role to query this monitor table.
- monWorkloadRaw – provides the raw workload statistics for each instance. You need not have the mon_role to query this monitor table.

Configuration Parameters

New configuration parameter for Adaptive Server Cluster Edition version 15.5 and later.

cluster redundancy level – The maximum number of instances that can fail simultaneously while allowing recovery to proceed concurrently with other activity. The cluster shuts down if the failed number of instances exceeds the maximum.

Functions

New functions for Adaptive Server Cluster Edition versions 15.5 and later.

- **xact_owner_instance** – returns the instance on which an external transaction is running, or 0.
- **xact_conmigrate_check** – determines whether a connection can process an external transaction.

Version 15.5

Adaptive Server 15.5 introduces in-memory and relaxed-durability databases, Backup Server support for IBM Tivoli Storage Manager, faster compression for backups, deferred name resolution for stored procedures, incremental data transfer, support for FIPS 140-2 password encryption, and new datatypes.

Adaptive Server 15.5 Feature and Platform Matrix

The feature and platform matrix shows feature availability for supported operating systems in Adaptive Server 15.5. A “Y” indicates the feature is supported for that platform.

Adaptive Server options	HP-UX Itanium 64-bit	HP-UX PA Risc 64-bit	IBM AIX 64-bit	Linux on Power 64-bit	Linux Opteron 64-bit	Linux x86 32-bit	Solaris 32-bit	Solaris 64-bit	Solaris Opteron 64-bit	Windows Opteron X64	Windows x86 32-bit
Security and directory services	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Cybersafe Kerberos							Y	Y			Y
Pluggable Authentication Module	Y		Y	Y	Y	Y	Y	Y	Y		
Fine-grained access control	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
LDAP server directory	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
LDAP user authentication	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Platform Native Kerberos							Y	Y			
Secure Sockets Layer	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
MIT Kerberos	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y
Encrypted columns, including fine-grained access control (FGAC)	Y	Y	Y		Y	Y	Y	Y	Y	Y	Y

Adaptive Server options	HP-UX Itanium 64-bit	HP-UX PA Risc 64-bit	IBM AIX 64-bit	Linux on Power 64-bit	Linux Opteron 64-bit	Linux x86 32-bit	Solaris 32-bit	Solaris 64-bit	Solaris Opteron 64-bit	Windows Opteron X64	Windows x86 32-bit
High availability	Y	Y	Y		Y	Y	Y	Y			Y
Partitions	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
In-memory databases	Y	Y	Y	Y	Y			Y	Y	Y	
Tivoli Storage Manager for Backup Server	Y		Y		Y			Y	Y	Y	
Active messaging	Y	Y	Y		Y	Y	Y	Y	Y	Y	Y
Enhanced Full-Text Search (EFTS)		Y	Y		Y	Y	Y	Y			Y
Features included in base Adaptive Server											
Cross-platform dump and load	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Job Scheduler	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Native XML	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
IPv6	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Java option	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Web Services	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Distributed transaction management	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Content management (external file support)	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Archived database access	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

In-Memory and Relaxed-Durability Databases

In-memory and relaxed-durability databases provide enhanced performance.

In-memory databases run entirely in a named cache (that is, in the Adaptive Server memory space), without using disk storage for data or logs. Because an in-memory database does not require I/O, its performance can be much better than a traditional, disk-resident database. In-memory databases are not designed for recovery: their transaction logs are written to the cache and not to disk, and any data changes are lost if the server is restarted. In-memory databases perform transactional logging for runtime rollback, and for other operations, such as firing triggers, deferred mode updates, replication, and so on.

Disk-resident databases perform writes to disk, and ensure that the transactional properties of atomicity, consistency, integrity, and durability (known as the ACID properties) are maintained. Durability refers to the persistence of transactions after they have committed. A traditional Adaptive Server database operates at full durability by writing its transaction log to disk when a transaction commits. This, along with data pages being written periodically to disk, ensures that all committed transactions are durable.

In-memory databases do not write data or log to disk, and trade the guarantee of transaction durability for performance improvements. In the event of a database failure, in-memory databases cannot be recovered. If your applications require data recoverability following a server failure or a normal shutdown, consider using a traditional Adaptive Server database.

With support for relaxed durability, Sybase extends the performance benefits of an in-memory database to disk-resident databases. A traditional disk-resident database guarantees transactional recovery from a server failure. Relaxed-durability databases trade the full durability of committed transactions for enhanced runtime performance for transactional workloads.

The performance benefits of in-memory and relaxed-durability databases include:

- An in-memory database operates entirely in-memory, so it does not wait for I/O.
- Improved buffer and user log cache management, so you need not create an overhead of user log cache flushes and buffer management when Adaptive Server performs concurrent updates to the same data.
- Runtime strategies that may avoid flushing task changes registered in the user-log cache to the transaction log when the transaction commits or aborts. These strategies reduce the contention on in-memory log pages.
- Support for minimally logged DML operations that use in-memory logging techniques improves the performance of large-volume DML operations.

Adaptive Server version 15.5 allows you to create these types of databases:

- Disk-resident databases with durability set to full (this is the default Adaptive Server database)

- User-defined disk-resident temporary databases
- In-memory user databases with durability set to no_recovery
- User-defined in-memory temporary databases with durability set to no_recovery
- Disk-resident relaxed-durability databases with durability set to no_recovery or at_shutdown

Adaptive Server supports **dump database** and **load database** for both in-memory and relaxed-durability databases. In addition, you can dump an in-memory database and load it into a disk-resident database, and vice versa. Adaptive Server supports template databases as a way to reinitialize the contents of nonrecoverable databases when the server restarts.

See the *In-Memory Database Users Guide*.

Faster Compression for Backups

New compression levels for dumping databases and transactions provide faster, more complete, and less CPU-intensive compression.

Two new compression levels have been added to the **dump database...compression=** and**dump transaction...compression=** commands: 100 and 101. Compression level 100 provides faster compression; level 101 provides better compression. Both new compression levels are less CPU-intensive than levels 0 – 9.

See the *System Administration Guide* and the *Reference Manual: Commands.*

Backup Server Support for the IBM® Tivoli® Storage Manager

Adaptive Server 15.5 supports IBM Tivoli Storage Manager (TSM) for storage management services in partnership with Backup Server. Support for TSM is a licensed Adaptive Server feature.

The TSM is a third-party client/server program that provides storage management services to licensed users. The Adaptive Server Backup Server supports tape drives and disk files as backup media. TSM works with Backup Server to enable a greater variety of backup media. You can store Adaptive Server backups on any backup media supported by TSM.

Adaptive Server handles the dump and load of databases and transactions to and from TSM; TSM handles storage and retrieval from the storage media. TSM provides storage management services, but you can administer all your Adaptive Server backup and restore operations, including queries for backup objects and deletion of backup objects, from Backup Server.

When you execute the **dump** or **load** commands, Backup Server invokes the Sybase interface with the TSM API, which provides communication with the TSM. When you use the **dump**

command, you specify an object name that is uniquely associated with the backup object. This object name is the same as the TSM object name, and should later be used to specify the same database or transaction dump for the load operation. In general, you can use the same options with the **dump** and **load** commands with TSM as you use with Backup Server when TSM is not configured.

Adaptive Server introduces these stored procedures to support the query and delete of backup objects from TSM:

- **sp_deletesmobj**– deletes some or all of the current server's backup objects from TSM.
- **sp_querysmobj** – retrieves a list of a server's backup objects.

See *Using Backup Server with IBM Tivoli Storage Manager*.

Deferred Name Resolution for User-Defined Stored Procedures

Deferred name resolution lets you create a stored procedure before referenced objects exist.

In versions of Adaptive Server earlier than 15.5, referenced objects were required to already exist before the stored procedure using them could be created. The deferred name resolution feature now allows objects, except for user-defined datatype objects, to be resolved when the stored procedure is executed for the first time.

This feature uses the **deferred name resolution** configuration parameter, which operates at the server level, or a new **set** command, **set deferred_name_resolution**, which operates at the connection level.

See the *Transact-SQL Guide*, the *System Administration Guide: Volume 1*, and the *Reference Manual: Commands*.

FIPS 140-2 Login Password Encryption

Adaptive Server 15.5 supports the FIPS 140-2 validated cryptographic module.

FIPS 140-2 certifed Certicom Security Builder GSE encrypts login passwords in a transmitted login packet, in memory and on disk, if you use the configuration parameter **FIPS login password encryption**:

```
sp_configure 'FIPS login password encryption', 1
```

Note: A Security and Directory Services license is required to enable this parameter. If the parameter is not enabled, the OpenSSL security provider performs login password encryption.

See the *Users Guide for Encrypted Columns*.

Incremental Data Transfer

Incremental data transfer lets you transfer data to Adaptive Server or other products.

The **transfer table** command allows you to transfer data incrementally, and, if required, to a different product. In versions of Adaptive Server earlier than 15.5, you could transfer only whole tables from one Adaptive Server to another.

Note: Adaptive Server enables the data transfer feature when you purchase, install, and register the in-memory database license.

Incremental data transfer:

- Lets you export data, including only the data that has changed since a prior transmission, from Adaptive Server tables that are marked for incremental transfer.
- Allows table data to be read without obtaining the usual locks, without guaranteeing any row retrieval order, and without interfering with other ongoing reads or updates.
- Lets you write selected rows to an output file (which can be a named pipe) formatted for a defined receiver: IQ (Sybase IQ), ASE (Adaptive Server Enterprise), bulk copy (**bcp**), or character-coded output. All selected rows are transmitted without encryption, and, by default, any encrypted columns within the row are decrypted before transmittal. The file to which you are writing must be visible to the machine on which Adaptive Server is running (the file can be an NFS file that Adaptive Server can open as a local file).
- Maintains a history of transmissions for eligible tables, and lets you remove transmission history when it is no longer wanted. Exports data from tables declared ineligible for incremental transfer, subject to certain restrictions.
- Transfers entire rows from indicated tables. You cannot currently select certain columns, select a partition within a table, or transfer results from SQL queries.

See Chapter 8, “Adding, Changing, Transferring, and Deleting Data,” in the *Transact-SQL Users Guide.*

bigdatetime and bigtime Datatypes

bigdatetime and *bigtime* provide microsecond precision.

Two new datatypes, *bigdatetime* and *bigtime*, provide a timestamp with microsecond precision that contains the year, month, day, hour, minute, second, and fraction of a second to six decimal places. A *bigdatetime* value requires 8 bytes of storage. A *bigtime* value contains the time of day, containing hour, minute, second, and fraction of a second. The fraction is stored to six decimal places. A *bigtime* value requires 8 bytes of storage.

See the *Adaptive Server Transact-SQL Users Guide*, the *Reference Manual: Building Blocks*, and *Java in Adaptive Server Enterprise*.

Creating and Managing tempdb Groups

Updates for **tempdb** groups for version 15.5

Adaptive Server allows you to create and manage user-created **tempdb** groups in addition to managing the default **tempdb** group. User-created **tempdb** groups can include other user-created temporary databases, and support application and login binding.

You cannot remove the system **tempdb** from the default temporary database group. You cannot add system **tempdb** to any other user-created **tempdb** group.

You can designate and administer user-created **tempdb** groups to contain only disk-resident or in-memory temporary databases. The server does not explicitly impose any such restriction, but by controlling the membership you can assign disk-only or in-memory-only **tempdb** groups to specific logins or applications.

System Changes for Adaptive Server Version 15.5

Adaptive Server 15.5 supports new and changed datatypes, functions, system procedures, commands, configuration parameters, monitoring tables, system tables, and utilities. New auditing options have also been added.

Datatypes

New *bigtime* and *bigdatetime* datatypes provide precision timestamp information.

Datatypes	Description
bigtime	A *bigtime* value includes the hour, minute, second, and fraction of a second. The fraction is stored to 6 decimal places.
bigdatetime	A *bigdatetime* value includes the year, month, day, hour, minute, second, and fraction of a second to 6 decimal places.

Two new functions return *bigtime* and *bigdatetime* values:

- **current_bigtime**
- **current_bigdatetime**

bigtime and *bigdatetime* can be used in these existing functions:

- **datepart**
- **datename**

- **datediff**
- **dateadd**

See the *Reference Manual: Blocks*.

Functions

New and changed functions in Adaptive Server 15.5

Table 23. New functions

Function	Description
db_attr	Returns the **durability**, **dml_logging**, and **template** settings for the specified database.
object_attr	Reports the table's current logging mode, depending on the session-specific, table-wide, and database-wide settings.
cache_usage	Returns cache usage for the cache to which the specified object is bound, as the percentage of the cache which is currently in use by all objects bound to that cache.
current_bigdatetime	Returns a *bigdatetime* value representing the current date and time with microsecond precision.
current_bigtime	Returns a *bigtime* value representing the current time with microsecond precision.

Table 24. Changed functions

Function	Description
datepart	Produces the specified *datepart* argument of the specified date as an integer.
datename	Produces the specified *datepart* as a character string.
datediff	Calculates the number of date parts between two specified dates or times.
dateadd	Adds an interval to a specified date or time.

See the *Reference Manual: Commands*.

System Stored Procedures

New and changed system stored procedures support Tivoli Storage Manager, and in-memory and relaxed-durability databases.

Table 25. New system stored procedures

System stored procedures	Description
sp_deletesmobj	Deletes backup objects from the TSM.
sp_querysmobj	Retrieves a list of backup objects from the TSM.

Table 26. Changed system stored procedures

System stored procedures	Description
sp_addsegment	Updated to manage space in in-memory databases.
sp_addthreshold	Updated to manage space in in-memory databases.
sp_bindcache	You cannot bind objects or databases to in-memory storage caches, and you cannot bind an in-memory database or objects in an in-memory database to any cache.
sp_cacheconfig	Creates, extends the size of, or drops, an in-memory storage cache.
sp_cachestrategy	The **prefetch** and **MRU** parameters do not apply to tables and indexes in in-memory databases.
sp_dbextend	Automatic database expansion is currently not supported for in-memory databases.
sp_deviceattr	The **directio** and **dsync device** attributes do not apply to in-memory devices.
sp_downgrade	Supports downgrading an Adaptive Server containing in-memory or relaxed-durability databases, or databases using templates or minimal logging.
sp_diskdefault	You cannot use **sp_diskdefault** to specify in-memory devices as a default devices.

System stored procedures	Description
sp_dropdevice	Drops an in-memory device created from an in-memory storage cache.
sp_dropsegment	Updated to manage space in in-memory databases.
sp_dropthreshold	Updated to manage space in in-memory databases.
sp_extendsegment	Updated to manage space in in-memory databases.
sp_help	Reports on properties, such as minimal logging attribute, for a table.
sp_helpcache	Displays properties of the in-memory storage cache, the in-memory database created on it, and details of free space on this cache.
sp_helpdb	Reports on database properties, such as durability, DML logging level, in-memory or not, use, if any, of a template database, or as a template database.
sp_helpdevice	Reports the in-memory device properties created from an in-memory storage cache.
sp_modifythreshold	Updated to manage space in in-memory databases.
sp_plan_dbccdb	Sets up **dbccdb** for checkstorage execution in an in-memory database.
sp_poolconfig	Large I/O buffer pools are not supported in an in-memory database.
sp_post_xpload	Supports cross-platform operations for in-memory databases.
sp_tempdb	Supports user-created temporary database groups, login or application bindings to temporary database groups and for in-memory databases.
sp_unbindcache, sp_unbindcache_all	You cannot unbind objects in or the in-memory database itself from the host in-memory storage cache.

See the *Reference Manual: Procedures*.

Commands

New and changed commands for Adaptive Server 15.5

Table 27. New commands

Command	Description
transfer table	Initiates an incremental table transfer. A new grant with grant option supports **transfer table**. It grants a specified user permission to transfer a specified table. `grant transfer table on table_name to user with grant option`

Table 28. Changed commands

Command	Change
alter database	Syntax added to support changing the durability of a database, the level of DML logging, a database's template.
alter table	Syntax added to support changing a table's logging mode for insert, update, and delete (DML) operations. Syntax added to support **transfer table**: `set transfer table [on | off]`
create database	Syntax added to create in-memory and relaxed-durability databases with durability set to full, no_recovery, or at_shutdown. Syntax added to specify DML logging level and template database, if any.
create table	Syntax added to specify DML logging level for tables in in-memory databases. Syntax added to support **transfer table**: `with transfer table [on | off]`
disk init	Syntax added to create in-memory data devices for in-memory databases.
dump database	Syntax added to support the Tivoli Storage Manager (TSM). The keyword **syb_tsm** invokes the Sybase interface with the TSM API (libsyb_tsm). `database_name to "syb_tsm::object_name"`

Command	Change
dump database ... compression=	Syntax added to support faster, less CPU-intensive compression levels 100 and 101.
dump transaction	Syntax added to support TSM. The keyword **syb_tsm** invokes the Sybase interface with the TSM API (libsyb_tsm). `database_name to "syb_tsm::object_name"`
dump transaction ... compression=	Syntax added to support faster, less CPU-intensive compression levels 100 and 101.

load database	Syntax added to support TSM. The keyword **syb_tsm** invokes the Sybase interface with the TSM API (libsyb_tsm).
	`database_name from "syb_tsm::`
	`[[-S source_server_name]`
	`[-D source_database_name]::]object_name"`

load transaction	Syntax added to support TSM. The keyword **syb_tsm** invokes the Sybase interface with the TSM API (libsyb_tsm).
	`database_name from "syb_tsm::`
	`[[-S source_server_name]`
	`[-D source_database_name]::]object_name"`

select into	Syntax added to specify the DML logging level for tables created by selecting into in-memory or relaxed-durability databases.

set	Adds:
	• **dml_logging** parameter for specifying the amount of logging for a session.
	• **deferred_name_resolution** for activating deferred name resolution at the connection level.
	• **builtin_date_strings** *number*
	Values are:
	  • 0 - if a string is given as an argument to a chronological system function, the server interprets it as a *datetime* value regardless of its apparent precision. This is the default.
	  • 1 - makes the server interpret the argument string as *bigdatetime*. This affects the result of chronological system functions.
	.

See the *Reference Manual: Commands*.

Configuration Parameters

Adaptive Server 15.5 introduces the **deferred name resolution** configuration parameter.

Configuration parameter	Description
deferred name resolution	Allows you to create procedures using deferred name resolution. Values are: • 0 – disables deferred name resolution. This is the default. • 1 – enables deferred name resolution.
builtin date strings	Values are: • 0 – causes string literals given to a chronological builtin function as an argument to be interpreted as a datetime type. • 1 – causes string literals given to a chronological builtin function as an argument to be interpreted as a bigdatetime type.

See the *System Administration Guide: Volume 1* and the *Transact-SQL Users Guide*.

Monitoring Tables

Adaptive Server 15.5 introduces the *monTableTransfer* monitoring table.

Monitoring table	Description
monTableTransfer	Provides historical transfer information for tables in the Adaptive Server active memory.

See the *Reference Manual: Tables*.

System Tables

New and changed system tables in Adaptive Server 15.5

Table 29. New system tables

System table	Description
spt_TableTransfer	Stores the results from table transfers.

Table 30. Changed system tables

System table	Change description
sysdevices	Lists the in-memory storage cache under the *name* and *phyname* columns. In-memory devices do not include a full path to the disk device, instead, they store the name of the cache on which the in-memory device has been created.
sysdatabases	Adds the *durability* column, which indicates the durability level of the database. The *durability* column has the *int* datatype. Its values are: • 1 – full • 5– at_shutdown • 6 – no_recovery

See the *Reference Manual: Tables*.

Utilities

New and changed utilities in Adaptive Server 15.5

Table 31. New utilities

Utility	Description
openssl	Performs all certificate management tasks implemented by certreq, certauth and certpk12. Sybase includes this binary as a convenience, and is not responsible for any issues incurred using the binary. See *www.openssl.org* for details.

Table 32. Changed utilities

Utility	Change
backupserver	Syntax change supports another verbosity level (V4) for the -V parameter. V4 displays all -V0 messages except “Connection from Server” messages printed for each connection event.

See the *Utility Guide*.

Auditing

Auditing options are added in support of in-memory and relaxed-durability databases, incremental data transfer, and deferred name resolution.

Table 33. Auditing enhancements

Audit option	Command or access to be audited	Event	Information in extrainfo
all, create	**transfer table**	136	Keywords or options: transfer_table
all, create	**alter table**	3	If **alter table** contains **set transfer table on**, Adaptive Server prints this to extrainfo: SET TRANSFER TABLE ON. If **alter table** contains **set transfer table off**, Adaptive Server prints this to extrainfo: SET TRANSFER TABLE OFF.
all, create	**create table**	12	If **create table** contains **with transfer table on**, Adaptive Server prints this to extrainfo: WITH TRANSFER TABLE ON. If **create table** contains **with transfer table off**, Adaptive Server prints this to extrainfo: WITH TRANSFER TABLE OFF.
all, create	**create database**	9	Keywords or options: inmemory
all, create	**alter database**	2	Keywords or options: inmemory
all, create	**create procedure**	11	Keywords or options: deferred_name_resolution

Version 15.0.3

Adaptive Server 15.0.3 introduces distributed transaction management, enhancements to the Java interface, virtually hashed tables, huge pages, updates to the Adaptive Server Plug-in, directions for upgrading during a High Availability configuration, and support for SQL statement replication.

SQL Statement Replication

Replication Server 15.2 supports SQL statement replication for Adaptive Server databases

Adaptive Server Enterprise 15.0.3 introduces SQL statement replication, which is supported by Replication Server 15.2 and later.

See the Replication Server documentation.

Security Enhancements

Adaptive Server version 15.0.3 introduces several new security enhancements.

LDAPS User Authentication Enhancement

Modifying the CA trusted root file no longer requires a server restart.

In earlier versions of Adaptive Server, if you modify the Certifying Authority (CA) trusted root file, you must restart Adaptive Server for the modifications to take effect. Adaptive Server version 15.0.3 and later supports modifications to the trusted root file, so that restarting the the server is unnecessary. A new subcommand, **reinit_descriptors**, which unbinds the LDAP server descriptors and reinitializes the user authentication subsystem.

Automatic LDAP User Authentication and Failback

The Adaptive Server housekeeping utility can automatically activate a failed LDAP server

Adaptive Server 15.0.3 provides support for a secondary LDAP server. Previously, after bringing a failed primary LDAP server online, it was necessary to activate the LDAP server manually, in order to authenticate new LDAP logins and move them to the primary LDAP server.

In versions 15.0.3 and later, a new chore has been added to Adaptive Server's housekeeping utility to activate an LDAP server automatically: **'set_failback_interval'**.

After you set the failback interval, the housekeeper task checks for failed LDAP servers each time it sweeps through its chores. When it finds a failed LDAP server, it attempts to activate the LDAP server when the failback time interval expires.

Login Mapping of External Authentication

Adaptive Server can map one unique mapping of an external user to an internal Adaptive Server login

When you configure an external authentication mechanism, if there is exactly one mapping of an external user to an internal Adaptive Server login, and if it is successfully authenticated, Adaptive Server updates the internal login's password to match the external user's password.

For example, under these conditions:

1. USER1 has an Adaptive Server login name of "user_ase" with password "user_password".
2. Another user has an LDAP login name of "user_ldap" with password "user_ldappasswd".
3. Adaptive Server has a one to one mapping for "user_ldap" to "user_ase".
4. User "user_ldap" logs in to Adaptive Server using password "user_ldappasswd". Adaptive Server updates the "user_ase" password to "user_ldapppasswd".

Using SSL to Specify a Common Name

Use a fully-qualified domain name for the SSL certificate common name

The server name specified in the directory service entry can be different from the common name the SSL server certificate uses to perform an SSL handshake. This allows you to use a fully-qualified domain name for the SSL certificate common name (for example, `server1.bigcompany.com`).

To add a common name to the interfaces file, use:

```
ase1
   master tcp ether host_name port_number ssl="CN='common_name'"
   query tcp ether host_name port_number ssl="CN='common_name'"
```

When clients use SSL to connect to an Adaptive Server that also uses SSL, the SSL filter is placed after the port number in the `interfaces` file. The directory service includes the common name, which you add either by using **dsedit** or a text editor.

sp_listener includes the **CN=common_name** parameter, which allows you to specify a common name for the SSL certificate.

Concurrent Kerberos Authentication

Adaptive Server can establish multiple Kerberos authentication sessions

Adaptive Server version 15.0.3 supports concurrent Kerberos authentication, whereas earlier versions used locking mechanisms during Kerberos authentication to protect internal data structures.

When there are concurrent logins using Kerberos authentication, Adaptive Server now establishes multiple Kerberos authentication sessions.

Version 15.0.3 also resolves an issue with concurrent login sessions, which may be blocked during Kerberos authentication. This concurrency issue occurs when you use prior versions of Adaptive Server with MIT version 1.3.x and 1.4.x Kerberos GSSAPI libraries.

Virtually Hashed Tables

Create virtually hashed tables to efficiently organize tables.

Note: Virtually hashed tables are available only on Linux pSeries.

You can perform hash-based index scans using nonclustered indexes or clustered indexes on data-only-locked tables. During this scan, each worker process navigates the higher levels of the index and reads the leaf-level pages of the index. Each worker process then hashes on either the data page ID or the key value in a separate hash table to determine which data pages or data rows to process.

A virtually hashed table can be a more efficient way to organize a table because it does not require a separate hash table. Instead, it stores the rows so that, using the hash key, the query processor can determine the row ID (based on the row's ordinal number) and the location of the data. Because it does not use a separate hash table to hold the information, it is called a "virtually" hashed table.

For systems that must make more efficient use of their central-processing unit (CPU), the virtually hashed table is a good option.

To create a virtually hashed table, specify the maximum value for the hash region using the **create table** command.

Huge Pages

Enable huge pages to use fewer pages to cover the physical address space.

Note: This feature is available only on Linux pSeries.

The CPU-Cache translation lookaside buffer (TLB) stores information about conversions from an virtual page address to the physical page address, and every byte access to physical memory requires a conversion (called a "cache miss"). Although these cache misses are very expensive, you can improve the TLB hits by enabling "huge pages."

Once enabled, huge pages use fewer pages to cover the physical address space, so the size of "book keeping" (mapping from the virtual to the physical address) decreases, requiring fewer entries in the TLB and improving the system performance.

Adaptive Server version 15.0.3 and later allocates shared memory using huge pages by default. However, if the system does not have enough huge pages—or is not configured for huge pages—Adaptive Server uses regular pages.

To enable huge pages, start Adaptive Server with traceflag 1653. Adaptive Server adjusts its shared memory up to the nearest multiple of 256MB.

Upgrading During a High Availability Configuration

After you have enabled high availability (HA), follow the upgrade instructions in the Adaptive Server installation guide for your platform

The instructions in this section supplement those in the *Installation Guide*.

Reinstalling System Stored Procedures

Reinstall the system stored procedures after enabling high availability

1. Disable HA in the primary server:

```
sp_companion secondary-server-name, 'drop'
sp_configure 'enable HA', 0
```

2. Disable HA in the secondary server:

```
sp_configure 'enable HA', 0
```

3. Restart the servers.
4. Run the `installmaster` script on both servers.
5. Enable the HA property on both servers:

```
sp_configure 'enable HA', 1
```

6. Restart both servers.
7. Run the `installhasvss` script on both servers. This script is located in `$SYBASE/$SYBASE_ASE/scripts`.
8. Reestablish companionship:

```
sp_companion [companion_server_name], configure     [,
with_proxydb]
```

Distributed Transaction Management (DTM)

Adaptive Serve automatically prevents SQL commands that are intended to execute inside a distributed transaction from executing outside it.

A distributed, or external, transaction is managed by an external transaction coordinator, such as XA Transaction Manager (TM).

In versions of Adaptive Server earlier than 15.0.3, user applications determined whether an external transaction was rolled back while executing DML commands. If Adaptive Server

implicitly aborted an external transaction without the application's knowledge, DML commands that would normally run inside this transaction might instead be executed inside an implicit transaction started by Adaptive Server. This behavior could result in inconsistent business data.

In versions 15.0.3 and later, Adaptive Server does not allow any DML commands to be executed on the connection attached to the external transaction until the transaction manager sends a detach request. The detach request indicates the end of a batch of commands intended for the external transaction.

In versions 15.0.3 and later, Adaptive Serve automatically prevents SQL commands that are intended to execute inside a distributed transaction from executing outside it. The user application no longer has to check the global variable before every command; when a transaction is implicitly aborted, an error message (3953) appears, saying "Cannot execute the command because the external transaction has been rolled back." This message disappears when a **detach transaction** command is issued.

Adaptive Server Plug-in Updates

The Adaptive Server Plug-in now runs on Sybase Central 6.00, and contains several new features.

The Adaptive Server Plug-in for Sybase Central manages various Adaptive Server Enterprise products. In versions earlier than 15.0.3, the Adaptive Server Plug-in ran on Sybase Central 4.3. In 15.0.3 the Adaptive Server Plug-in runs on Sybase Central 6.00. These features are new to version 15.0.3, Sybase Central 6.00:

- A Search tool helps you find objects displayed by plug-ins.
- The Connection Profile Description, Import, and Export options allow you to add a text description to a profile connection.
- There is better support for Windows Vista.

These features are new to the version 15.0.3 Adaptive Server plug-in.

- You can create objects by selecting the Add icon from a context-sensitive toolbar.
- Stored procedures and SQLJ procedures are located in the Procedures folder.
- Scalar functions, or user-defined functions, are now supported.
- Utilities items are now accessible from the menu on the context-sensitive toolbar.

DBISQL11, which was previously shipped as part of Adaptive Server Plug-in, is now a separate product, version 11.0, and includes these features:

- The number of multiple result sets is no longer limited to 10.
- The login dialog for Adaptive Server now retains and displays the last five connected server names.

- DBISQL11, or interactive SQL, now supports connection favorites, which are similar to connection profiles.
- The SQL statements pane now contains line numbers.
- The Results pane now shows using **select all**, **insert**/**update**/**delete** SQL statements, and sorting and generating, from selected rows.

The Java Interface

Java in Adaptive Server now supports third-party JRE and JVM components such as J2SE.

Adaptive Server version 15.0.3 lets you plug in off-the-shelf Java Runtime environment (JRE) and JVM components, such as J2SE, to Adaptive Server. This Adaptive Server Java framework is called the pluggable component interface (PCI), which includes pluggable component adaptors (PCAs). Any JVM configured for Adaptive Server is called a "plug-in."

The Adaptive Server Java framework allows you to build on the Java solution in Adaptive Server versions 15.0.2 and earlier without losing significant functionality. Any Java applications you developed using Adaptive Server versions earlier than 15.0.3 should run seamlessly with Java applications you create using the framework in versions 15.0.3 and later.

After you configure Adaptive Server to run with the PCI, you can include any standard JVM that supports Java 6 or later. This separates your Java applications from Adaptive Server, allowing you to change or upgrade your Java applications independent of Adaptive Server and to take advantage of new Java functionality as it becomes available.

See *Java in Adaptive Server Enterprise* for a complete description of the new Adaptive Server Java interface.

System Changes for Adaptive Server Version 15.0.3

Adaptive Server 15.0.3 introduces new and changed functions, system stored procedures, commands, configuration parameters, and system tables.

Functions

Adaptive Server 15.0.3 introduces the **password_random** and **pssinfo** functions.

Function	Description
password_random	Generates a pseudorandom password that satisfies the global password complexity checks defined on Adaptive Server.

Function	Description
pssinfo	Returns information from the process status structure, with an option that retrieves the transaction isolation level of any spid.

See the *Reference Manual: Commands*.

System Stored Procedures

New and changed system stored procedures in Adaptive Server 15.0.3

Table 34. New system stored procedures

Stored procedure	Description
sp_tabsuspectptn	A range-partitioned table on character-based partition keys can become suspect after a sort-order change, and hash-partitioned tables can become suspect after a cross-platform dump load.
sp_jreconfig	This is a Java stored procedure.

Table 35. Changed System Stored Procedures

Procedure	Change
sp_ldapadmin	Supports the new parameters **set_failback_interval** and **reinit_descriptors**.
sp_addserver	Supports the **filter** parameter to add a remote server for remote procedure calls.
sp_passwordpolicy	Supports the **validate password options** parameter.
sp_pciconfig	Supports Java in the database.
sp_sysmon	Supports additional counters.

See the *Reference Manual: Procedures*.

Commands

Adaptive Server 15.0.3 introduces changes to the **create table** and **update statistics** commands.

Command	Description of change
create table	The table you create with this option is available only to **BCP IN** and **'alter table unpartition'** operations.
update statistics	Resets the data change counters for global nonclustered indexes.

See the *Reference Manual: Commands*.

Configuration Parameters

New configuration parameters for Adaptive Server 15.0.3

Procedure	Change
enable pci	This is a Java configuration parameter.
maximum nesting level	The maximum nesting level has been increased to 100.
mnc_full_index_filter	Prevents Adaptive Server from considering noncovered indexes that do not have a limiting search argument at the server level if there is: • A column in the index • A predicate that does not have a histogram
pci memory size	This is a Java configuration parameter

See the *System Administration Guide: Volume 1*.

Monitoring Tables

New and changed monitoring tables for Adaptive Server 15.0.3

Table 36. New monitoring tables

Table	Description
monSQLRepActivity	Provides statistics for all open objects on DML statements replicated using SQL statement replication.
monSQLRepMisses	Provides statistics for replicated operations for which SQL statement replication was not used. The *threshold*, *querylimitation* and *configuration* columns indicate the number of times that one of these factors prevented SQL statement replication for the object

Table 37. Changed monitoring tables

Monitoring table	Description of change
monSysStatement, *monSysPlanText*, and *monSysSQLText*	The values of the columns *BatchID*, *ContextID*, *ProcedureID*, and *PlanID* have been modified.
monSysStatement	Supports two new columns: *ProcNestLevel* and *StatementNumber*.

See the *Reference Manual: Tables*.

System Tables

Adaptive Server 15.0.3 adds new columns to several system tables.

Table	Changes
sysqueryplans	New columns: • *dbid, int null* • *qpdate, datetime null* • *sprocid, int null* • *hashkey2,int null* • *key1, int null* • *key2, int null* • *key3, int null* • *key4, int null* **Note:** These columns are reserved for future use.
sysprocedures	New column *qp_setting varbinary(6) null* **Note:** This column is reserved for future use.
sysprocesses	New column *clientport* • Displays client port numbers for client processes • Displays 0 for system processes • Datatype: unsigned *smallint*
sysservers	The column *srvnetname* has changed from *varchar(32)* to *varchar(255)*.

See the *Reference Manual: Tables*.

Version 15.0.2

Adaptive Server 15.0.2 introduces many new features and enhancements. They include enhancements to security, encrypted columns, and performance. New features include archive database access, deferred compilation, eager and lazy aggregate processing, and user-defined SQL functions.

Encrypted Columns

Adaptive Server Enterprise 15.0.2 provides enhancements to encrypted columns

The new features:

- Protect data from administrator. You can protect keys and encrypted columns with your own password to ensure privacy of data against the power of the DBO or System Administrator.
- Maintain application transparency using key copies protected by login passwords. That is, you can create key copies and assign them to individual users. Users can encrypt their key copies using their login passwords. Once a key copy is associated with a login password, users do not have to supply the key encryption password when they access data encrypted with the key.
- Provide for key recovery. You can recover access to a key after losing a password. The key owner sets up a recovery key copy, which can later be used to reencrypt the key after losing the password.
- Return a default value for users without decrypt permission. You can create or alter a table to allow **select** statements to return specified default values for users who do not have decrypt permission. This allows you to run existing applications and reports without generating a permission error, while keeping private data secure against unauthorized users. Reports generated by unauthorized users do not reveal the encrypted data.
- Restrict automatic decrypt permissions. When the **restricted decrypt permission** configuration parameter is enabled, the System Security Officer explicitly grants decrypt permission, restricting access to data. When **restricted decrypt permission** is enabled:
  - Table owners are not implicitly granted decrypt permission. The schema owner does not have automatic and implicit access to user data, even in systems that rely on the system encryption password to access the keys.
  - Only users with the sso_role can grant decrypt permission. The **with grant** option is supported for decrypt permission.
  - Implicit access through ownership chains across view and tables or procedures and tables is restricted.

- Adds datatypes. You can encrypt these additional datatypes: *date*, *time*, *datetime*, *smalldatetime*, *money*, *smallmoney*, *big int*, *unsigned big int*, *bit*, *unichar* and *univarchar*.

Archive Database Access

Validate or selectively recover data from a database dump (an “archive”) by making the dump appear as if it is a traditional, read-only database (an "archive database")

Unlike a traditional database, an archive database uses the actual database dump as its main disk storage device, with a minimum amount of traditional storage to represent new or modified pages that result from the recovery of the database dump. A database dump already contains the images of many (if not most) of the database pages, therefore, an archive database can be loaded without having to use Backup Server to transfer pages from the archive to traditional database storage. Consequently, the load is significantly faster than a traditional database.

Archive database access enables a variety of operations to be performed directly on a database dump.

An archive database does not have to be a complete copy of the original database. Depending on the optimization used when dumping the database using **sp_dumpoptimize**, an archive database may be fully populated (every page in the database is in the database dump), or partially populated (only allocated pages are stored in the database dump).

Because the database dump is presented as a read-only database, a database administrator can query it using familiar tools and techniques such as:

- Running database consistency checks on the most recent copy of a dump made from a production database. These checks can be offloaded to a different server to avoid resource contention in the production environment. If resources are not a concern, the archive can be directly checked in the same server in which it was created. Verification on the archive provides the assurance needed prior to performing a restore operation.
- If the integrity of a database dump is in question, loading it into an archive database can be a quick test for success, and therefore a good tool to identify the appropriate database dump that should be used to restore a traditional database.
- Object-level restoration from the database dump. Lost data is recovered using **select into** to copy the to-be-restored rows from the table within the archive database. The **select into** operation can be performed either directly in the server hosting the archive database, or by using Component Integration Services proxy tables if the archive database is available on a different server than that of the object requiring restoration.

In addition, transaction logs can be loaded into an archive database, thereby providing assurance that the same load sequence can be applied when performing a restore operation.

Finding Slow-Running Queries

Adaptive Server 15.0.2 introduces new **set** commands that collect information about slow-running queries.

These parameters for the **set** command enable you to collect diagnostic information about poorly running queries without having to previously enable **showplan** or other investigatory parameters:

- **tracefile** – saves diagnostics to a trace file
- **show_sqltext** – displays SQL text
- **export_options** – retains session settings

Deferred Compilation

Adaptive Server 15.0.2 introduces deferred compiling. Using deferred compilation, the optimizer can compile stored procedural statments that reference real runtime values.

The optimizer can now perform a runtime compilation of procedural statements that reference local variables and temporary tables, so that the query is optimized with real runtime values, instead of magic numbers.

- Adaptive Server uses deferred compilation for queries that reference local variables and parameters in search clauses, queries where a join is used with a temporary table created in the same procedure, and queries where a subquery references a temporary table.
- A statement qualified for deferred compilation is compiled at the first execution of the stored procedure. Those statements that are not executed the first time the stored procedure is invoked, for example statements omitted due to **IF** clauses, are not compiled until a subsequent execution of the stored procedure actually executes these statements.
- Once a conditional statement is compiled, whether at the first or at a subsequent procedure execution, it is integrated into the query plan and is not recompiled.
- Queries that reference procedure parameters previously were compiled and optimized only with the the value of those parameters upon entry to the stored procedure. In versions 15.0.2 and later, these statements are optimized with the parameter values they had when the query was first executed. If the parameter value alters during the course of the stored procedure execution, the current value is used in optimization.
- You can switch off deferred compilation by starting the server with the global switch -T7730.

Case-Insensitive Sort Orders for Chinese and Japanese Character Sets

This section describes case-insensitive sort orders for these Chinese and Japanese character sets:

- EUC-GB
- GB-18030
- CP-936
- EUC-JIS
- SJIS
- DECKANJI

Table 38. Sort orders available for Simplified Chinese and Japanese

Language or script	Character sets	Sort orders
Simplified Chinese	EUC-GB, GB-18030, CP936	General purpose case-insensitive dictionary ordering
Japanese	EUCJIS, SJIS, DECKANJI	General purpose case-insensitive dictionary ordering

Statistical Aggregate Functions

Adaptive Server 15.0.2 introduces statistical aggregate functions to compute variance and standard deviation

Aggregate functions summarize data over a group of rows from the database. The groups are formed using the **group by** clause of the **select** statement.

Simple aggregate functions, such as **sum**, **avg**, **max**, **min**, **count_big**, and **count** are allowed only in the **select** list and in the **having** and **order by** clauses as well as the **compute** clause of a **select** statement. These functions summarize data over a group of rows from the database.

Adaptive Server Enterprise now supports statistical aggregate functions, which permit statistical analysis of numeric data. These functions include **stddev**, **stddev_samp**, **stddev_pop**, **variance**, **var_samp**, and **var_pop**.

These functions, including **stddev** and **variance**, are true aggregate functions in that they can compute values for a group of rows as determined by the query's **group by** clause. As with other basic aggregate functions such as **max** or **min**, their computation ignores null values in the input. Also, regardless of the domain of the expression being analyzed, all variance and standard deviation computation uses IEEE double-precision floating-point standard.

If the input to any variance or standard deviation function is the empty set, then each function returns as its result a null value. If the input to any variance or standard deviation function is a single value, then each function returns 0 as its result.

Standard Deviation and Variance

Learn about the new statistical aggregate functions and their aliases.

- **stddev_pop** (also **stdevp**) – standard deviation of a population.Computes the population standard deviation of the provided value expression evaluated for each row of the group (if **distinct** was specified, then each row that remains after duplicates have been eliminated), defined as the square root of the population variance.
- **stddev_samp** (also **stdev**, **stddev**) – standard deviation of a sample. Computes the population standard deviation of the provided value expression evaluated for each row of the group (if **distinct** was specified, then each row that remains after duplicates have been eliminated), defined as the square root of the sample variance.
- **var_pop** (also **varp**) – variance of a population. Computes the population variance of value expression evaluated for each row of the group (if **distinct** was specified, then each row that remains after duplicates have been eliminated), defined as the sum of squares of the difference of value expression from the mean of value expression, divided by the number of rows in the group.
- **var_samp** (also **var**, **variance**) – variance of a sample. Computes the sample variance of value expression evaluated for each row of the group (if **distinct** was specified, then each row that remains after duplicates have been eliminated), defined as the sum of squares of the difference from the mean of the value expression, divided by one less than the number of rows in the group.

Eager and Lazy Aggregation

Aggregate processing summarizes large amounts of data with an aggregated value.

Aggregate processing is one of the most useful operations in DBMS environments. It summarizes these values:

- The minimum, maximum, sum, or average value of a column in a specified set of rows
- The count of rows that match a condition
- Other statistical functions

In SQL, aggregate processing is performed using the aggregation functions **min()**, **max()**, **count()**, **sum()**, and **avg()**, and **group by** and **having** clauses. The SQL language implements two aggregate processing types, *vector aggregation* and *scalar aggregation*. A **select-project-join** (SPJ) query illustrates these two types of aggregate processing:

```
select r1, s1
from r, s
where r2 = s2
```

Vector and Scalar Aggregation

Adaptive Server 15.0.2 supports vector and scalar aggregation

In vector aggregation, the SPJ result set is grouped on the **group by** clause expressions, and then the **select** clause aggregation functions are applied to each group. The query produces one result row per group:

```
select r1, sum (s1)
from r, s
where r2 = s2
group by r1
```

In scalar aggregation, there is no **group by** clause and the entire SPJ result set is aggregated, as a single group, by the same **select** clause aggregate functions. The query produces a single result row:

```
select sum (s1)
from r, s
where r2 = s2
```

Improved Performance for Data Insertion

Adaptive Server 15.0.2 optimizes performance of data insertion

These 15.0.2 features can enhance performance for data insertion:

- Fast **bcp** can copy data into tables with nonclustered indexes or triggers, improving Adaptive Server's performance for inserting huge volumes of data.
- Adaptive Server version 15.0.2 includes a separate user log cache (ULC) for the session's temporary database, so multidatabase transactions that include a single user database and the session's temporary database do not require ULC flushes when the users switch between the databases or if all of the following conditions are met:
  - Adaptive Server is currently committing the transaction.
  - All the log records are in the ULC
  - There are no post-commit log records.

  The configuration option, **session tempdb log cache size**, allows you to configure the size of the ULC, helping to determine how often it needs flushing.
- When Adaptive Server splits an index or data page, it moves some rows from the original pages to the newly created page. The operation of moving the rows is not logged. Adaptive Server version 15.0.2 uses asynchronous writes to disk that do not require the server to block as it waits for the write to complete. Adaptive Server version 15.0.2 uses these asynchronous writes automatically and requires no configuration on your part.
- Improved throughput of tempdb transactions
- Adaptive Server version 15.0.2 provides post-commit optimization. The server performs two scans of the log: the first scan looks for data page deallocation and unreserved pages, the second scan looks for log page deallocation. These scans are an internal optimization,

transparent to users, and are performed automatically; you cannot switch the scans on or off.

With post-commit optimization, Adaptive Server remembers the “next” log page (in the backward direction) containing these log records. During the post-commit phase, Adaptive Server moves to the “next” page requiring post-commit work after processing records from a page. In a concurrent environment, where many users log their transactions to *syslogs* at the same time, post-commit optimization can improve the performance of post commit operation by avoiding reads or scans of unnecessary log pages.

Using Asynchronous Writes During a Page Split

When Adaptive Server splits an index or data page, it moves some rows from the original pages to the newly created page. The operation of moving the rows is not logged. Adaptive Server 15.0.2 uses asynchronous writes to ensure consistency.

To ensure both consistency and durability, Adaptive Server must satisfy these conditions:

- Adaptive Server writes the new page to disk before writing the modified page (with the rows removed) to disk. This ensures that Adaptive Server can restore the previous version of the page if the transaction is undone. Adaptive Server can find these rows on the new page and move them back to the old page even if the rows are missing in the old page, and their row contents were not logged.
- The new page reaches the disk before the transaction commits, which ensures that Adaptive Server cannot loose the committed data. If the transaction was committed, Adaptive Server is not required to redo the transaction for the new page, which would be impossible since the movement of the rows was not logged. In the case of undo, the new page’s allocation is backed out; there's no page pre-image to restore.

Previous versions of Adaptive Server ensured these two conditions were met by synchronously writing the new page to disk. However, because the server could block until the synchronous write returned, this caused a degradation in performance.

Adaptive Server version 15.0.2 uses asynchronous writes to disk that satisfy the conditions described above and do not require the server to block as it waits for the write to complete.

Adaptive Server version 15.0.2 uses these asynchronous writes automatically and requires no configuration on your part.

Improving Throughput of tempdb Transactions

Earlier versions of Adaptive Server flushed the data pages and single log records (SLRs) because crash recovery was not supported for *tempdb* or any databases not requiring recovery.

SLRs are log records that force a flush of the user log cache (ULC) to *syslogs* immediately after the record is logged. SLRs are created for OAM modifications, and Adaptive Server creates log records affecting allocation pages in a mixed log and data database as SLRs.

- For regular databases, a ULC containing SLRs is flushed immediately to avoid any undetected deadlocks caused during buffer "pinning". Avoiding a ULC flush for SLRs reduces log semaphore contention, improving the performance.

  A ULC flush avoids the deadlock caused by buffer pinning. Because Adaptive Server does not pin the buffers for databases that do not need recovery, it avoids this deadlock and does not have to flush the ULC for SLRs.
- For databases that require recovery, Adaptive Server flushes dirty pages to disk during the checkpoint. This ensures that, if Adaptive Server crashes, all the committed data is saved to disk. However, for databases which do not require recovery, Adaptive Server supports a runtime rollback, but not a crash recovery. This allows it to avoid flushing dirty data pages at a checkpoint and improves performance.
- Adaptive Server does not support write ahead logging on databases that do not require recovery. Write-ahead logging guarantees that data for committed transactions can be recovered by "redoing" the log (reperforming the transactions listed in the log), and "undoing" the changes done by aborted or rolled back transactions to maintain database consistency. Write-ahead logging is implemented by the "buffer pinning" mechanism. Since Adaptive Server does not ensure write-ahead logging on databases not needing recovery, it does not pin buffers for these databases, so it can skip flushing the log when it commits a transaction.

Post-commit Optimization

Adaptive Server version 15.0.2 performs two scans of the log: the first scan looks for data page deallocation and unreserved pages, the second scan looks for log page deallocation. These scans are an internal optimization, transparent to users, and are performed automatically; you cannot switch the scans on or off.

Previous versions of Adaptive Server:

- Used three scans of the log record after a committed transaction, one each for page deallocation, unreserved pages, and log deallocation.
- Performed the backward scan of log pages using page linkages. Some of the pages did not have log records requiring post-commit work.

With post-commit optimization, Adaptive Server remembers the "next" log page (in the backward direction) containing these log records. During the post-commit phase, Adaptive Server moves to the "next" page requiring post-commit work after processing records from a page. In a concurrent environment, where many users log their transactions to *syslogs* at the same time, post-commit optimization can improve the performance of post commit operation by avoiding reads or scans of unnecessary log pages.

The optimization does not show up in any diagnostics.

Changes to the Query Processor

These modifications to the query processor describe behavior changes that can affect query plan selection.

Deferred Compilation

The query processor for Adaptive Server version 15.0.2 defers the optimization of statements in a stored procedure until it executes the statement.

Deferring the optimization of statements benefits the query processor because the values for local variables are available for optimization for their respective statements.

Earlier versions of Adaptive Server used default guesses for selectivity estimates on predicates using local variables.

Non-binary Character Set Histogram Interpolation

Adaptive Server version 15.0.2 allows selectivity estimates to have the same accuracy as the binary character set.

In earlier versions of Adaptive Server, only the default binary character set benefited from histogram interpolation, which is used to estimate the selectivity of range predicates. For all other character sets, Adaptive Server made a selectivity estimate of 50 percent for a histogram cell. This typically required Adaptive Server to use a large number of histogram cells for character column histograms to reduce the error associated with this estimate.

Adaptive Server version 15.0.2 allows selectivity estimates to have the same accuracy as the binary character set, without requiring an excessive number of histogram steps. This benefits queries like the following with range predicates:

```
select * from t1 where charcolumn > "LMC0021" and
charcolumn <= "LMC0029"
```

If ranges specified falls into the same histogram cell, Adaptive Server can much more accurately estimate this selectivity.

Expression Histogramming Selectivity Estimates

Adaptive Server version 15.0.2 applies histogramming estimates to single column predicates if the histogram exists on the column.

Earlier versions of Adaptive Server used default “guesses” for selectivity estimates.

Adaptive Server version 15.0.2 applies histogramming estimates to single column predicates if the histogram exists on the column. This results in more accurate row estimates, and improves the join order selection for query plans.

In this example, if the expression is very selective, it may be better to place table *t1* at the beginning of the join order:

```
select * from t1,t2 where substring(t1.charcol, 1, 3)
= "LMC" and t1.a1 = t2.b
```

Viewing Current Optimizer Settings

A new fake table *sysoptions* and a new stored procedure **sp_options** let you view optimizer settings.

To let you easily view optimizer settings, Adaptive Server version 15.0.2 introduces:

- *sysoptions* – a new fake table that stores information about each **set** option, its category, and its current and default settings. *sysoptions* also contains a bitmap containing further detailed information about the option.
- **sp_options** – a new stored procedure that shows option values.

New Security Features

New security features in Adaptive Server 15.0.2

PAM Support in 64-bit Adaptive Server on AIX

Adaptive Server 15.0.1 supports PAMUA on AIX 5.2.

Adaptive Server version 15.0.2 on AIX 5.2 64-bit supports Pluggable Authentication Module-based User Authentication (PAMUA). Contact your support representative to make sure you have the latest patch for PAM available on your IBM host.

To enable PAMUA in 64-bit Adaptive Server 15.0.2 on AIX 5.2, you must supply the PAM module in `/etc/pam.conf` file. For example:

```
ase auth required /usr/lib/security/pam_aix
```

Global Login Triggers Set Automatically

For Adaptive Server version 15.0.2 and later, any exportable option set or unset in a login trigger takes effect in the login process when the server starts.

Adaptive Server versions 15.0.1, 12.5.4, and earlier required that you start Adaptive Server with trace flag 4073 to enable the options for a login trigger.

Any exportable option set or unset in a login trigger now takes effect in the login process when the server starts

To disable the new behavior, execute **set export_options off** inside the login trigger.

SSL Support

Adaptive Server 15.0.2 supports SSL on Windows 2003 x64 Enterprise Edition.

Adaptive Server version 15.0.2 supports Secure Sockets Layer (SSL) on Windows 2003 X64 Enterprise Edition. Windows 2003 X64 implements SSL functionality using OpenSSL libraries.

SSL functionality is the same as Adaptive Servers on other platforms. It supports the same cipher suites and pre-defined lists as shown in the *System Administration Guide*. OpenSSL libraries for Adaptive Server on Windows 2003 X64 communicates with SSLPlus clients and servers using the same cipher suite names used in SSLPlus. The cipher suite names conform to the Transport Layer Security (TLS). TLS is an enhanced version of SSL 3.0, and is compatible with the SSL version 3.0 Cipher Suites.

Improved Password Security

Adaptive Server 15.0.2 improves password security.

Adaptive Server 15.0.2 adds a higher level of security to the existing password protection mechanisms through:

- Stronger encryption for passwords sent across the network
- Stronger encryption for passwords stored in *syslogins* (on disk) and in memory
- Managing login and password use with new time stamp data and additional account reporting

Auditing Enhancements

Adaptive Server version 15.0.2 introduces two additional auditing security features.

Hiding System Stored Procedure and Command Password Parameters

System stored procedure and command passwords can be replaced with asterisks in audit records.

When auditing is configured and enabled, and the **sp_audit** option **'cmdtext'** is set, system stored procedure and command password parameters are replaced with a fixed length string of asterisks in the audit records contained in the audit logs.

For example, execute:

```
sp_password 'oldpassword', 'newpassword'
```

When auditing is enabled and **sp_audit cmdtext** is set, the results in output are similar to:

```
sp_password '******', '******'
```

This protects passwords from being seen by other with access to the audit log.

Monitoring Failed Login Attempts

Monitor failed logins attempts with the **login_locked** audit option.

The new audit option **login_locked** and the event **Locked Login (value 112)** record when a login account is locked due to exceeding the configured number of failed login attempts. This event is enabled when audit option **login_locked** is set. To set **login_locked**, enter:

```
sp_audit "login_locked","all","all","ON"
```

If the audit tables are full and the event cannot be logged, a message with the information is sent to the errorlog.

The hostname and network IP address are included in the audit record. Monitoring the audit logs for the **Locked Login** event (112) helps to identify attacks on login accounts.

High Availability Considerations

The Adaptive Server 15.0.2 password security changes affect High Availability (HA).

HA configuration

The primary and companion servers must have equivalent "**allow password downgrade**" values before they are configured for HA. A new quorum attribute "**allow password downgrade**" has been added to check whether the value of "**allow password downgrade**" is same on both primary and secondary servers. This HA advisory check succeeds when the value for the quorum attribute is the same, and HA advisory check fails when the values are different.

Changed password behavior in syslogins with HA

After upgrading to Adaptive Server 15.0.2 and successful configuring HA, on the first connection to the primary server, the password of the user login is updated on both the primary and companion servers. This synchronizes the login password on primary and companion with the same on-disk encryption format. This is done to avoid password reset or locking when the "**allow password downgrade**" period ends as described in **sp_passwordpolicy** and with password downgrade to earlier Adaptive Server 15.0 versions with **sp_downgrade**. By synchronizing the password encryption format, the login passwords can continue to be used without being reset or locked by **sp_passwordpolicy** or **sp_downgrade**.

Installing and Editing Monitoring Tables

Adaptive Server version 15.0.2 includes new installation and editing features for monitoring tables.

- Installing monitoring table - you no longer need to run the `installmontables` script to install the monitoring tables.

Versions of Adaptive Server earlier than 15.0.2 required you to run the `installmontables` script to install the monitoring tables. Adaptive Server version 15.0.2 includes the monitor tables installation in the `installmaster` script.

- Remotely accessing and editing monitoring tables - Sybase provides `installmontables` as a sample script showing how to remotely access monitoring tables. Run `installmontables` to view the instructions for editing.

Monitoring Tables for the Statement Cache

Two new monitoring tables let you analyze the contents of the statement cache.

The Adaptive Server statement cache stores SQL text of ad-hoc **update**, **delete** and **select** statements and other statements likely to be reused. When the statement cache is enabled, these statements are converted into lightweight procedures and their plans are saved for reuse. When a new statement is issued, Adaptive Server searches the statement cache for a plan to reuse. If Adaptive Server finds a plan to reuse, it avoids recompiling the statement, leading to performance enhancements.

The introduction of literal parameterization in Adaptive Server version 15.0.1 allows it to recognize queries that are the same except for differences in literal values, saving recompiling costs while using statement cache. In addition to performance benefits, literal parameterization leads to enormous space reduction while storing the metrics and statements in the cache.

Adaptive Server version 15.0.2 introduces two new monitoring tables that allow you to easily analyze the contents of the statement cache:

- *monStatementCache* provides a summary snapshot of the statement cache.
- *monCachedStatement* shows detailed information about each statement cached

The columns in each table allow two attributes, “counter” if the column has a counter value, and “reset” if the column can be reset using mechanisms like **sp_sysmon**.

Row-Level Locking for System Tables

Adaptive Server version 15.0.2 uses row-level locking on system tables to enhance performance.

Versions of Adaptive Server earlier than 15.0.2 used exclusive table locks on system tables while executing data definition language (DDL) and utility commands. The set of system tables Adaptive Server locked depended on the type of DDL operation you executed. If another DDL running concurrently tried to take a conflicting exclusive table lock on the same system table, this DDL had to wait to acquire the lock on any system catalogs. These DDL operations were executed serially.

This methodology impeded performance in temporary databases, where their DDL activity is very high also, and consequently their catalog contention is very high. This limited the Adaptive Server throughput for applications using temporary tables.

Adaptive Server version 15.0.2 uses row-level locking to resolve these issues:

- System-table contention, caused a bottleneck for many DDLs and utilities.
- *tempdb* contention. Because the system tables are locked at the row level, Adaptive Server 15.0.2 eliminates *tempdb* contention.
- Shared or exclusive table-level locks while executing DDLs and utilities. Earlier versions converted most system tables to data-only locking (DOL), but still created shared or exclusive table-level locks while executing DDLs and utilities. Using row-level locks for system tables eliminates this contention.

  Adaptive Server sets intent locks on catalogs only, which removes potential contention (An intent lock indicates that page-level or row-level locks are currently held on a table.).
- DDLs and utilities blocking each other. Adaptive Server 15.0.2 allows DDLs and utilities to run in parallel.

Earlier versions of Adaptive Server used table locks to achieve system catalog synchronization. Adaptive Server 15.0.2 uses intent locks for table-level synchronization and row locks for row-level synchronization. Earlier releases of Adaptive Server locked the entire system catalog while performing operations on the object, so a single lock request was made. However, Adaptive Server version 15.0.2 requests locks for all applicable rows while performing operations on the object if there are multiple rows corresponding to an object in a system catalog.

This change means that Adaptive Server 15.0.2 requests more locks to perform the same operation than earlier releases, and increases the number of lock resources the system needs. Consequently, you may need to change the **number of locks** configuration option after you upgrade Adaptive Server.

The xmltable() Function

xmltable() creates a SQL table from elements in an XML table.

xmltable() extracts a sequence of multivalued elements from an XML document, and assembles a SQL table of those elements. A single call to **xmltable()** replaces a Transact-SQL loop that performs multiple calls to **xmlextract()** on each iteration. **xmltable()** is invoked as a derived table (a parenthesized subquery specified in the **from** clause of a different SQL query). Calling **xmltable()** is equivalent to executing a single **xmlextract()** expression for each row of the table generated by **xmltable()**.

xmltable() is a generalization of **xmlextract()**. Both functions return data extracted from an XML document that is an argument in the function. The differences are:

- **xmlextract()** returns the data identified by a single XPath query.

- **xmltable()** extracts the sequence, or row pattern, of the data identified by an XPath query, and extracts from each element of that sequence the data identified by a list of other XPath queries, the column patterns. It returns all the data in a SQL table.

Relocated Joins

Relocation joins permits joins between local and remote tables to locate to remote server.

Adaptive Server version 15.0.2 introduces relocated joins, which allow joins between local and remote tables to be relocated to a remote server. The remote system executes the join using a dynamically created proxy table referring back to the local table. With the remote system executing the join, a significant amount of network traffic is avoided.

User-Defined SQL Functions

Adaptive Server 15.0.2 introduces user-defined SQL functions.

Use **create function** to create user-defined functions, and **drop function** to remove a user-defined function.

You can include these elements in a scalar function:

- **declare** statements to define data variables and cursors that are local to the function.
- Assigned values to objects local to the function (for example, assigning values to scalar and variables local to a table with **select** or **set** commands).
- Cursor operations that reference local cursors that are declared, opened, closed, and deallocated in the function.
- Control-of-flow statements.
- **set** options (only valid in the scope of the function).

Adaptive Server does not allow **fetch** statements in a scalar function that return data to the client. You cannot include :

- **select** or **fetch** statements that returns data to the client.
- **insert**, **update**, or **delete** statements.
- Utility commands, such as **dbcc**, **dump** and **load** commands.
- **print** statements
- Statement that references **rand**, **rand2**, **getdate**, or **newid**.

You can include **select** or **fetch** statements that assign values only to local variable.

instead of Triggers

Use **instead of** triggers to override default triggering actions.

Views are commonly used to separate logical database schema from physical schema. **instead of** triggers can be defined on a view to replace the standard action of an **update**, **insert**, or **delete** statement. The **instead of** trigger allows all views, including those that are not otherwise updatable, to be updated.

instead of triggers are special stored procedures that override the default action of a triggering statement (**insert**, **update**, and **delete**), and perform user-defined actions.

The **instead of** trigger, like the **for** trigger, executes each time a data modification statement executes on a specific view. A for trigger fires after an **insert**/**update**/**delete** statement on a table, and is sometimes called an **after** trigger. A single **instead of** trigger can apply to one specific triggering action:

```
instead of update
```

It can also apply to multiple actions, in which the same trigger executes all the actions listed:

```
instead of insert,update,delete
```

Like **for** triggers, **instead of** triggers use the logical **inserted** and **deleted** tables to store modified records while the trigger is active. Each column in these tables maps directly to a column in the base view referenced in the trigger. For example, if a view named V1 contains columns named C1, C2, C3, and C4, the **inserted** and **deleted** tables contain the values for all four columns, even if the trigger modifies only columns C1 and C3. Adaptive Server automatically creates and manages the **inserted** and **deleted** tables as memory-resident objects.

instead of triggers allow views to support updates, and allow implementation of code logic that requires rejecting parts of a batch, while allowing other parts to succeed.

An **instead of** trigger is fired only once per data modification statement. A complex query containing a **while** loop may repeat an **update** or **insert** statement many times, firing the **instead of** trigger each time.

System Changes for Adaptive Server Version 15.0.2

Changes to commands, functions, utilities, system procedures, system tables, configuration parameters, functions, and global variables in Adaptive Server 15.0.2

Trace Flags

New trace flags in Adaptive Server 15.0.2

- 15340 enables server wide, no matter advanced_aggregation
- 15341 disables server wide, no matter advanced_aggregation

Commands

Changed commands in Adaptive Server 15.0.2

Table 39. Changed commands

Command	Description of change
disk init, **disk reinit**	Display a warning message if you attempt to create a block device on a platform that Sybase recommends that you not use block device. Sybase recommends that you use block devices as a database device only on the HP-UX, Windows, and Linux platforms.
create proxy table, create table at remote server, and **alter table.**	Do not support SQL UDF.
disk init, **disk resize**	When *skip_alloc* is set to be true, it allows users to avoid initialization of pages with zeros. The default of *skip_alloc* is false. It is supported for devices created on non-Windows file systems and on Windows raw systems.
dump transaction	**dump transaction** can now include **string** and **char_variable** parameters. However, you cannot supply string and **char_variable** options to execute **begin transaction**, **commit**, **connect to**, **declare cursor**, **rollback**, **dbcc**, **use** or nested **execute** commands.

See the *Reference Manual: Commands*.

Changes to the set Command

Adaptive Server 15.0.2 introduces several changes to the **set** command.

set command change
set advanced_aggregation enables and disables advanced aggregation at the session level.

set command change
set switch allows you to set trace flags and switch names locally and server-wide.
The compile-time behavior has changed for some abstract plan **set** parameters when you use them to create stored procedures or run them in Transact-SQL batches.
The **set** command can include **string** and **char_variable** parameters.

Utilities

Adaptive Server 15.0.2 supports enhancements to the **ddlgen** utility.

Change	Description
ddlgen prompts for password.	In version prior to 15.0.2, failing to include the **-P** password parameter returned an error. **ddlgen** now prompts for the password.
ddlgen supports both pre-15.0.2 and 15.0.2 encryption.	Sample syntax: `ddlgen -Usa -P -Sserver -TEK -NSampleKeysDB.dbo.ssn_key`
ddlgen without the **-XOD** flag	Two things can happen: • If users do not specify a password when the encryption key is created, **ddlgen** generates DDL with no password. • If users specify a password when the encryption key is created, **ddlgen** generates a default password.
ddlgen with the **-XOD** flag	Generates a system encryption password.
Key copy support	**ddlgen** can generate DDL for key copies and for the base key.

See the *Utility Guide*.

System Stored Procedures

New and changed system procedures in Adaptive Server 15.0.2

Table 40. New system stored procedures

System stored procedure	Description
sp_downgrade	Validates readiness for downgrade to an earlier 15.0.x release. Also downgrades the system catalog changes Adaptive Server 15.0.2 modified.
sp_spaceusage	Reports the space usage for a table, index, or transaction log and estimates the amount of fragmentation for tables and indexes in a database. The estimates are computed using an average row-length for data and index rows, and the number of rows in a table. You can archive the space usage and fragmentation data for future reporting and trends analysis. **sp_spaceusage** supports a number of actions, including **help**, **display**, **archive** and **report**, to indicate the current Adaptive Server space usage.

Table 41. Changed system stored procedures

System stored procedure	Description of change
sp_autoformat	Now accepts columns of datatypes *int* (*smallint*, *bigint*, *tinyint*, *unsigned int*), *numeric*, *money*, *date/time*, and *float*, *real*, and *double* precision.
sp_changedbowner	Changes the owner of a database. You can now execute it with either sa_role or sso_role privileges. The owner of thresholds for that database is also changed to the specified user.
sp_checksource	Encrypts the text of user-defined functions.
sp_configure	Now displays non-default value settings.
sp_depends	Checks for any object dependencies for user-defined functions.
sp_deviceattr	Displays a warning message if the **dsync** option is disabled for a database device on a file system.

System stored procedure	Description of change

sp_displaylogin	includes these changes:
	• supports both a wildcard expression and a server user ID, and displays matching logins:
	`sp_displaylogin ['user_id' | '[loginame | wildcard]'`
	• **user_id** – user ID (*suid*) of the user whose login you are displaying.
	• **wildcard** – wildcard character used for search purposes.
	Displays the login account for the user with a suid of 56:
	`sp_displaylogin '56'`
	Displays the login account information for all users whose logins begin with "st":
	`sp_displaylogin 'st%'`

sp_droplogin	When **sp_droplogin** is unable to drop a login due to the existence—in any database—of a user in *sysusers* referencing the login **suid**, the names of databases in which the references are found are now displayed in the error message.
sp_help	Displays information about user-defined functions.
sp_helpdevice	The *description* column of **sp_helpdevice** displays information about the device type. The device type is one of: `raw device`, `block device`, or `file system device`.
sp_helprotect	The new option, **permission_name**, in **sp_helprotect** provides information (grantor name, grantee name, table/column name, and grantability) for any specific permission granted in a given database.
sp_hidetext	Encrypts the text for user-defined functions.
sp_locklogin	The **lock** option to **sp_locklogin**, when used with a value for **number of inactive days**, locks inactive accounts that have not authenticated within that period. The following example locks all login accounts that have not authenticated within the past 60 days.
sp_modifylogin account	A new value for the **'max failed_logins'** option indicates that the failed login count in the *syslogins* column login count, is updated whenever an authentication failure occurs, but that the account is not locked.

System stored procedure	Description of change
sp_modifystats	Allows the System Administrator, or any user with permission to execute the procedure and update statistics on the target table, to modify the density values of columns in *sysstatistics*
sp_monitorconfig	Enhanced to create a table to hold the result set, if the user passes a table name for **result_tabl_name** that does not already exist.
sp_passwordpolicy	• The **set** and **clear** commands in **sp_passwordpolicy** are now audited, through audit event 115, "Password Administration." • Additional syntax: `sp_passwordpolicy "enable last login updates", "allow password downgrade" "regenerate keypair", "expire login passwords", "[login_name	wildcard]" "expire role passwords", "[role_name	wildcard]" "expire stale login passwords", "datetime" "expire stale role passwords", "datetime" "maximum failed logins", -1`
sp_fixindex	Now works on a set of indexes rather than on a single index. **sp_fixindex** rebuilds the data layer if the target table has a placement or clustered index (it reclaims the unused space in the data layer while working on the placement or clustered index of a system table).
sp_sendmsg	In previous releases, the maximum length for a message sent with this system procedure was 255 characters. For Adaptive Server release 15.0.2, the maximum length of a **sp_sendmsg** message is 4096 characters.
sp_who	A new column, *tempdbname*, displays temporary database names of all active sessions.
sp_helptext	Reports the text of user-defined functions, and introduces the **numlines** and **printops** parameters.
sp_ldapadmin	Introduces new parameters: **set_max_ldapua_desc**, **set_num_retries**, and **set_log_interval**.
sp_monitor	Enhances the **event** and **help** parameters.
sp_tempdb	Introduces the **show** and **who** parameters.

See the *Reference Manual: Procedures*.

System Tables

New and changed system tables in Adaptive Server 15.0.2.

Table 42. New system tables

System table	Description
sysoptions	*sysoptions* is a fake table queried by **sp_options**. The columns names are case-sensitive.

Table 43. Changed system tables

System table	Description of change
sysquerymetrics	Adaptive Server version 15.0.2 increases the number of metrics shared among user IDs. The change reduces the number of entries in *sysquerymetrics* (a view of *sysqueryplans*), and automatically aggregates the metrics for identical queries across different user IDs.
syscolumns	Adds a status bit to the *status2* column that indicates an encrypted column has a *decrypt_default* attached to it: 0x00001000 (4096) – column has a decrypt default
sysobjects	The *type* column of *sysobjects* includes an entry of "*DD*" for each object that has a decrypt default.

sysaudits	Changes include:
	• The *Alter Encryption Key* audit event name is changed to *AEK As/Not Default*
	• Adaptive Server release 15.0.2 supports these audit events and numbers:
	  • 118 – AEK Modify Encryption
	  • 119 – AEK Add Encryption
	  • 120 – AEK Drop Encryption
	  • 121 – AEK Modify Owner
	  • 122 – AEK Key Recovery

System table	Description of change

sysattributes	*sysattributes* includes these changes:
	• Adds two classes:
	  • Class 31 *allow password downgrade* – when set to 1, *allow password downgrade* enables special handling of login passwords for compatibility with Adaptive Server release 15.0 and earlier.
	  • Class 32 *enable last login updates* – when set to 1, *enable last login updates* enables system tables to store the date of the last login.
	• *sysattributes* includes information about default decrypt. These are the changes to the columns:
	  • *attribute* – specifies a default decrypt on an encrypted column with a value of 1 (DECRYPT-DEFAULT_ID) for objects with a type of *EC* and a class of 25.
	  • *object* – includes the decrypt default ID.
	  • *object_info_1* – includes the table ID for a table whose encrypted column defines the decrypt default.
	  • *object_info2* – specifies the **colid** of the encrypted column that includes the decrypt default.

System table	Description of change

sysencryptkeys	Changes to *sysencryptkeys* includes
	• New types:
	  • *EK_KEYCOPY* – 0x0010,
	  • *EK_KEYBASE* – 0x0020
	  • *EK_RECOVERY* – 0x0040
	• New status bits:
	  • *EK_KEYRECOVERY*(0x00000004) – keys encrypted for lost password protection.
	  • *EK_LOGINACCESS*(0x00000008) – key encrypted for login access
	  • *EK_LOGINPASS* (0x00000010) – key encrypted with login password
	  • *EK_USERPWD*(0x00000100) – keys encrypted with user-encryption passwords
	• Changes to the description for the *uid* column – user access or key recovery row. *uid* contains the user ID (*uid*) identifying the database user associated with current row.
	Previous versions of Adaptive Server did not use the *uid* column.

syslogins	In previous releases the maximum length of the *password* column was 30 bytes. In Adaptive Server release 15.0.2, the maximum length of the *password* column is 128 bytes. New columns are: *lastlogindate*, *crdate*, *locksuid*, *lockreason*, and *lockdate.*
syssrvroles	In previous releases the maximum length of the *password* column was 30 bytes. In Adaptive Server release 15.0.2, the maximum length of the *password* column is 128 bytes.

See the *Reference Manual: Tables*.

Configuration Parameters

New and changed configuration parameters in Adaptive Server 15.0.2

Table 44. New configuration parameters

Configuration parameter	Description
enable merge join	Enables or disables merge joins at the server level.

Configuration parameter	Description
cost of a logical io	Specifies the cost of a single logical IO.
cost of a physical io	Specifies the cost of a single physical IO.
cost of a cpu unit	Specifies the cost of a single CPU operation.
enable encrypted columns	Enables and disables the encrypted columns feature when encrypted columns is licensed.
max online q engines	Required for MQ series, **max online q engines** specifies the maximum number of Q engines allowed online.
metrics elap max	If the elapsed time of the query is less than the value of **metrics elap max**, then the metrics associated with the query are not written to the system tables. **enable metrics capture** must be on.
metrics exec max	If the execution time of the query is less than the value of **metrics exec max**, then the metrics associated with the query are not written to the system tables. **enable metrics capture** must be on.
metrics lio max	If the logical IO time of the query is less than the value of **metrics lio max**, then the metrics associated with the query are not written to the system tables. **enable metrics capture** must be on.
metrics pio max	If the physical IO time of the query is less than the value of **metrics pio max**, then the metrics associated with the query are not written to the system tables. **enable metrics capture** must be on.
min pages for parallel scan	Lets you scan larger queries in parallel.
not password encryption read	Requires all incoming login authentication requests using Adaptive Server authentication to encrypt the login password when transmitted on the network.
number of Q engines at startup	Required for MQ series, specifies the maximum number of Q engines you can have online.
prod-consumer overlap factor	Affects optimization. Adaptive Server changes the group by algorithm, and you cannot use **set statistics IO** with parallel plans.
send doneinproc tokens	Replaces the **dbcc tune** option **doneinproc**.

Table 45. Changed configuration parameters

Configuration parameter	Description of change
max async i/os per engine	The default value has changed from 2147483647 to 1024.
maximum failed logins	The new -1 value indicates that the failed login count in the *syslogins* column *logincount* is updated whenever an authentical failure occurs, but that the account is not locked.
print deadlock information	Adds a new parameter value of 2, which lets you print a summary of deadlock information to the error log.

See the *Reference Manual: Tables*.

Functions

New and changed functions in Adaptive Server 15.0.2

Table 46. New functions

Function	Description
authmec()	Returns the authentication method used for a logged-in server process ID session.
index_name()	Returns the name of an index, when you specify the ID of the index and the database, and the object on which the index is defined.
hashbytes()	Produces a fixed-length, hash value expression.

Table 47. Changed functions

Function	Description of change
used_pages()	In all-pages-locked tables with clustered indexes, **used_pages()**is now passed only the used pages in the data layer, for a value of **indid = 0**. When **indid = 1** is passed, the used pages at the data layer and at the clustered index layer are returned.
When a function is created, Adaptive Server checks to see if it is a SQL UDF or a SQLJ UDF. If it is a SQLJ UDF, Adaptive Server checks for “sa” permissions. If it is a SQL function, Adaptive Server checks for **create function** privileges.	

Global Variables

Adaptive Server 15.0.2 introduces the *@@lastlogindate* global variable.

Table 48. New global variables

Global variable	Description
@@lastlogindate	Global T-SQL variable **@@lastlogindate** is available to each user login session. A *datetime* datatype, its value is the *lastlogindate* column for the login account before the current session was established. This variable is specific to each login session and can be used by that session to determine the previous login to the account. If the account has not been used previously or "**sp_passwordpolicy 'set', enable last login updates**" is 0, then the value of **@@lastlogindate** is NULL.

Table 49. Changed global variables

Global variable	Description of change
@@opttimeoutlimit	Previous version of Adaptive Server user documenation listed **@@opttimeout** as a server global variable that displays the current optimization timeout limit for query optimization. This is incorrect. The actual name of the global variable that displays the current optimization timeout limit for query optimization is **@@opttimeoutlimit**

Version 15.0.1

Adaptive Server 15.0.1 introduces several enhancements to abstract plans and lets you automatically convert literal values in SQL queries to parameter descriptions.

Changes to Abstract Plans

Adaptive Server 15.0.1 provides enhancements to abstract plans.

The enhancements include:

- The abstract plan syntax has been extended to allow several query level setting that were previously available only at the session level.
- Adaptive Server accepts both the h_join and hash-join keywords in the extended abstract plan syntax.
- The **set** command supports the *opt criteria* parameter to turn on and off the current optimization goal setting.

New Query-Level Settings

The abstract plan syntax, used by Adaptive Server Enterprise to force the query plan chosen by the optimizer, has been extended to allow several query level setting that were previously available only at the session level.

The optimization criteria are handled at the session level by the following **set** statements:

```
set
    nl_join|merge_join|hash_join|..
    0|1
```

The **use ...** abstract plan syntax has been extended to accept any number of use forms before the abstract plan derived table. Previously, the optgoal and opttimeout could not be in the same abstract plan with a derived table. For example this statement would need to be separate from a optgoal statement in a query:

```
select ...
    plan
    "(use opttimeoutlimit 10) (i_scan r)"
```

However, with Adaptive Server 15.0.1, you can include several statements in the same abstract plan in two ways:

- By using several **use** statements, for example:

```
select ...
       plan
       "(use optgoal allrows_dss)
            (use nl_join off) (...)"
```

- By placing several items within one **use** form, for example:

```
select ...
       plan
       "(use (optgoal allrows_dss) (nl_join off)) (...)"
```

At the query level, use the optimization goal (**opt_goal**) or timeout (**opttimeout**) setting with the **use ...** abstract plan syntax. At the session level, use these settings with the **set plan ...** syntax:

- Optimization goal
- Optimization timeout

Operator Name Alignment for the Abstract Plan and the Optimizer Criteria

Adaptive Server accepts both the h_join and hash_join keywords in abstract plans.

The names of algorithms differ between their usage in abstract plans and in the **set** command. For example, a hash join is called h_join in abstract plans, but is called hash_join in the **set** command. Adaptive Server accepts both keywords in the extended abstract plan syntax. For example:

```
select ...
    plan
    "(h_join (t_scan r) (t_scan s))"
```

is equivalent to:

```
select ...
    plan
    "(hash_join (t_scan r) (t_scan s))"
```

and:

```
select ...
    plan
    "(use h_join on)"
```

and:

```
select ...
    plan
    "(use hash_join on)"
```

When a table abstract plan is present, it takes precedence:

```
select
from r, s, t
...
    plan
        "(use hash_join off)(h_join (t_scan r)
            (t_scan s))"
```

The query uses the hash_join for **r** and **s** scans; but for the join with **t** it does not use hash_join as specified by the use abstract plan form, since it was not specified in the table abstract plan.

Extending the Optimizer Criteria Set Syntax

The **set** command supports the *opt criteria* parameter to turn on and off the current optimization goal setting.

The **set** <*opt criteria*> statement, with a 0 or 1 syntax, accepts on/off/default, where default indicates that you are using the current optimization goal setting for this optimization criteria.

See the *Reference Manual: Commands*).

Literal Parameterization

Adaptive Server version 15.0.1 lets you automatically convert literal values in SQL queries to parameter descriptions (similar to variables).

In earlier versions of Adaptive Server, two queries that were identical except for one or more literal values resulted in the statement cache storing two separate query plans, or two additional rows in *sysqueryplans*. For example, the query plans for these queries were stored separately, even though they are almost identical:

```
select count(*) from titles where total_sales > 100
select count(*) from titles where total_sales > 200
```

Adaptive Server version 15.0.1 allows you to automatically convert literal values in SQL queries to parameter descriptions (similar to variables). A new **sp_configure** option supports this feature, which is called **enable literal autoparam**.

To enable or disable **enable literal autoparam** server-wide, use:

```
sp_configure "enable literal autoparam", [0 | 1]
```

Where 1 automatically converts literals to parameters, and 0 disables the feature. The default is 1.

You can set literal parameterization at the session level with the **set** command:

```
set literal_autoparam [off | on]
```

System Changes for Adaptive Server Version 15.0.1

Functions

Adaptive Server 15.0.1 introduces three new functions.

Function	Description
isdate	Determines whether an input expression is a valid *datetime* value.
isnumeric	Determines if an expression is a valid *numeric* datatype.
partition_object_id	Displays the object ID for a specified partition ID and database ID

See the *Reference Manual: Commands*.

Configuration Parameters

New and changed configuration parameters for Adaptive Server 15.0.1

Table 50. New configuration parameters

Configuration parameter	Description
startup delay	Controls when RepAgent is started during the server start. By default, RepAgent starts at the same time as Adaptive Server. Adaptive Server writes a message to the error log stating the wait time.
enable literal autoparam	Enables and disables literal server-wide parameterization.
cis idle connection timeout.	configures Adaptive Server to check for CIS connections to any remote server that have been unused longer than the specified number of seconds. Adaptive Server deletes the unused connections and reallocates their resources. Although the number you specify is in seconds, the housekeeper task wakes up at most once a minute, so idle connections may be idle for much longer than the configured value. Adaptive Server does not drop idle connections if a transaction is active on the connection, and reestablishes the connection automatically if the user executes any command that accesses the connection.

Configuration parameter	Description
sproc optimize timeout limit	Specifies the amount of time Adaptive Server can spend optimizing a system procedure as a fraction of the estimated execution time.

Table 51. Changed configuration parameters

Configuration parameter	Description of change
optimization timeout limit	The range of values available for **optimization timeout limit** has changed. With version 15.0.1, it is 0 - 1000. A value of 0 indicates no optimization timeout.
max parallel degree	If **max parallel degree** is set to 1 (enabled), Adaptive Server forces serial query execution and the optimizer may select plans with a higher parallel degree than if it is disabled.
number of worker processes	If you have not configured **number of worker processes** for a sufficient number of threads from the worker thread pool, Adaptive Server adjusts query plans at runtime to use fewer worker threads. If Adaptive Server cannot adjust the queries at run-time, the queries recompile serially. However, **alter table** and **execute immediate** commands are aborted if they do not have sufficient worker threads.

See the *System Administration Guide: Volume 1*.

Commands

Adaptive Server 15.0.1 introduces syntax and other changes to **alter table**, **create index**,**create existing table** , **update statistics**, and the **set** command.

Table 52. Changed commands

Table	Description of change
alter table	Use the **alter table** command to drop one or more list or range partitions. You cannot use **alter table** to drop a hash or round-robin partition. The syntax is: `alter table table_name drop partition partition_name [, partition_name]...`
create index	When you create a unique local index on range-, list-, and hash-partitioned tables, the index key list is a superset of the partition-key list.
create existing table	Includes syntax to determine whether an RPC uses the current or a separate connection: `create existing table (<column_list>) EXTERNAL [non_transactional |transactional] PROCEDURE at 'location'` • `non_transactional` – a separate connection is used to execute the RPC. • `transactional` – the existing connection is used to execute the RPC. The default behavior is transactional.
update statistics	Adaptive Server 15.0.1 adds the ability to run **update statistics** on a global index. `update table statistics table_name [partition data_partition_name] [index_name [partition index_partition_name]]` Because running **update table statistics** incurs the I/O cost of running **update statistics**, use **update statistics** to generate both column and table statistics. You can create, and then drop, a global index to generate global statistics.

Table 53. New set command options

New set command options	Description
set literal_autoparam on|off	Enables and disables literal parameterization at the session level.
set opttimeoutlimit	The range of values for **opttimeoutlimit** has been changed to 0 – 4000, with 0 indicating no optimization limit.
set index_union on | off	When enabled, **set index_union** limits the scan of a table with an **or** clause. Index unions (also known as an **or** strategy) are used for queries that contain **or** clauses. For example: `select * from titleauthor where au_id = "409-56-7008" or title_id = "PC8888"` If you have enabled **index_union**, this example uses an index on *au_id* to find the row IDs (RIDs) of all *titleauthor* tuples with `au_id = "409-56-7008"`, and uses an index on *title_id* to find the RIDs of all *titleauthor* tuples with `title_id = "PC8888"`. Adaptive Server then performs a union on all RIDs to eliminate duplicates. The resulting RIDs are joined with a **RidJoin** to access the data tuples. If **index_union** is disabled, Adaptive Server does not use an index union strategy in a query to limit the table scan. Instead, it uses other access paths on the table (in the example above, it would use a table scan for table *titleauthor*), and applies the **or** clause as a filter in the scan operator.

See the *Reference Manual: Commands*.

Monitoring Tables

Adaptive Server 15.0.1 introduces two new monitoring tables: *monProcedureCacheMemoryUsage* and *monProcedureCacheModuleUsage.*

Monitoring table	Description
monProcedureCacheMemoryUsage	Has one row for each procedure cache allocator. An allocator is identified by an allocator ID, which is internal to Adaptive Server.
monProcedureCacheModuleUsage.	Has one row for each module that allocates memory from procedure cache. A module, which is identified with a module ID, is a functional area classification internal to Adaptive Server procedure cache management.

See the *Performance and Tuning Guide: Monitoring and Analyzing* and the *Reference Manual: Tables*.

Version 15.0

Adaptive Server 15.0 introduces support for data partitions, row-locked system catalogs, an enhanced query processor, large identifiers, computed columns scollable cursors, new datatype support, XML enhancements, support for interactive SQL, enhancements to SySAM and the Adaptive Server Plug-in, user-defined Web service support, and security enhancements.

Partition Support

Partitioning is useful in managing large tables and indexes by dividing them into smaller, more manageable pieces. Partitions, like a large-scale index, provide faster and easier access to data.

Partitions are database objects and can be managed independently. You can, for example load data, and **create index** cannot be done at a partition level.. Yet partitions are transparent to the end user, who can select, insert, and delete data using the same commands whether the table is partitioned or not.

Adaptive Server 15.0 supports horizontal partitioning, in which a selection of table rows can be distributed among partitions on different disk devices. Individual table or index rows are assigned to a partition according to a semantic or to a round-robin partitioning strategy.

Semantic partitioning strategies use the data values in specified, key columns in each row to determine the partition assignment of that row. The round-robin partitioning strategy assigns rows randomly without reference to data values.

Partitioning strategies are:

- *Hash partitioning* (semantic) – a system-supplied hash function determines the partition assignment for each row.
- *List partitioning* (semantic) – values in key columns are compared with sets of user-supplied values specific to each partition. Exact matches determine the partition assignment.
- *Range partitioning* (semantic) – values in key columns are compared with a user-supplied set of upper and lower bounds associated with each partition. Key column values falling within the stated bounds determine the partition assignment.
- *Round-robin partitioning* – rows are assigned randomly to partitions in a round-robin manner so that each partition contains a more or less equal number of rows. This is the default strategy.

You can:

- Create partitions when you create a table or index using the **create table** and **create index** commands.
- Alter a table's partitioning strategy using the **alter table** command.
- Add a partition to an existing table with **add partition**.
- You can use partitioning to expedite the loading of large amounts of table data—even when the table eventually will be used as an unpartitioned table.

Row-Locked System Catalogs

Many system catalogs can now use a datarows locking scheme.

Adaptive Server version 15.0 converts most system catalogs to a datarows locking scheme. These system catalogs continue to use allpages locking scheme:

- Materialized tables such as *syslocks* and *sysprocesses*. These tables are generated during run-time and their locking schemes are irrelavent for concurrency.
- *sysmessages* and *sysusermessages*, which are read-only tables
- Auditing tables in *sybsecurity*, which are write-once and read many times.

Adaptive Server's internal upgrade process converts the system table locking schemes during an installation, upgrade, or load upgrade.

Because DDLs in Adaptive Server release 15.0 use the same table-level locks as 12.5.x and earlier versions, there is no concurrency improvement when you run DDLs.

Query Processor

Adaptive Server 15.0 provides an enhanced query processor

The Adaptive Server version 15.0 query processor is self-tuning, requiring fewer interventions than earlier versions. This version of Adaptive Server has less reliance on worktables for materialization between steps since the engine supports data flow between steps. However, more worktables could be used in cases where Adaptive Server determines that hash and merge operations are effective.

New features include support for:

- Both vertical and horizontal parallelism for query processing
- Improved index selection, especially for joins with OR clauses and joins and search arguments (SARGs) with mismatched but compatible datatypes
- More efficient algorithms
- Improved costing, using join histograms for joins with data skews in joining columns
- Improved query plan selection that enhances performance through:

    - New index union and index intersection strategies for queries with **and**/**or** predicates on different indexes
    - On-the-fly grouping and ordering using in-memory sorting and hashing for queries with **group by** and **order by** clauses
    - Cost-based pruning and timeout mechanisms that use permutation search strategies for large, multi-way joins, and for star and snowflake schema joins
- Improved problem diagnosis and resolution using:
    - Searchable XML format trace outputs
    - Diagnostic output from new **set** commands
- Joins involving a large number of tables
- Data and index partitioning, which are especially beneficial for very large data sets

Partitioning is the basic building block for parallelism.

Adaptive Server release 15.0 provides roundrobin partitioning. Round robin partitionin is equivalent to the 12.5 style of partitioning. During the upgrade to Adaptive Server release 15.0, all existing partitioned tables are unpartitioned and automatically converted to 1-way round robin partitioned tables.

Large Identifiers

Adaptive Server 15.0 introduces expanded limits for delimited identifiers

There are new limits for the length of object names or identifiers: 255 bytes for regular identifiers, and 253 bytes for delimited identifiers. The new limit applies to most user-defined identifiers including table name, column name, index name and so on. Due to the expanded limits, some system tables (catalogs) and built-in functions have been expanded.

For variables, “@” count as 1 byte, and the allowed name for the variable i 254 bytes.

Computed Columns

Computed columns and function-based indexes provide easier data manipulation and faster data access.

Computed columns provide easier data manipulation and faster data access by allowing you to create computed columns, computed column indexes, and function-based indexes.

- Computed columns – defined by an expression, whether from regular columns in the same row, functions, arithmetic operators,or path names.
- Indexes on computed columns, or computed column indexes – indexes that contain one or more computed columns as index keys.
- *Function-based indexes – indexes that contain one or more expressions as index keys.*

- Deterministic property – a property assuring that an expression always returns the same results from a specified set of inputs.

Computed columns and function-based indexes similarly allow you to use an expression or a function as the basis for a more complex function.

Differences Between Computed Columns and Function-Based Indexes

Computed columns and function-based indexes differ in some respects.

- A computed column provides both shorthand for an expression and indexability, while a function-based index provides no shorthand; it allows you to index the expression directly.
- A computed column can be either deterministic or nondeterministic, but a function-based index must be deterministic. “Deterministic” means that if the input values in an expression are the same, the return values must also be the same.

Differences Between Materialized and Not Materialized Computed Columns

Computed columns can be materialized or not materialized.

- Columns that are materialized are preevaluated and stored in the table when base columns are inserted or updated. The values associated with them are stored in both the data row and the index row. Any subsequent access to a materialized column does not require reevaluation; its preevaluated result is accessed. Once a column is materialized, each access to it returns the same value.
- Columns that are not materialized are also called virtual columns; virtual columns become materialized when they are accessed. If a column is virtual, or not materialized, its result value must be evaluated each time the column is accessed. This means that if the virtual computed column is expression-based on, or calls a nondeterministic expression, it may return different values each time you access it.You may also encounter run-time exceptions, such as domain errors, when you access virtual computed columns.

Scrollable Cursors

Adaptive Server Enterprise 15.0 supports both scrollable and nonscrollable cursors.

“Scrollable” means that you can scroll through the cursor result set by fetching any, or many, rows, rather than one row at a time; you can also scan the result set repeatedly. You must use Transact-SQL or JDBC to declare a scrollable cursor, and you must have the query engine provided in Adaptive Server 15.0 or later. A scrollable cursor allows you to set the position of the cursor anywhere in the cursor result set for as long as the cursor is open, by specifying the option **first**, **last**, **absolute**, **next**, **prior**, or **relative** in a **fetch** statement.

To fetch the last row in a result set, enter:

```
fetch last [from] <cursor_name>
```

Or, to select a specific row in the result set, in this case the 500th row, enter:

```
fetch absolute 500 [from] <cursor_name>
```

“Insensitive” or “semi-sensitive” refers to the extent to which data changes from outside the cursor are visible to the cursor. A cursor can be semi-sensitive but not scrollable.

All scrollable cursors are read-only. All **update** cursors are nonscrollable.

unitext Datatype Support

Adaptive Server 15.0 supports the vaiable-length *unitext* datatype.

The variable-length *unitext* datatype can hold up to 1,073,741,823 Unicode characters (2,147,483,646 bytes). You can use *unitext* anywhere you use the *text* datatype, with the same semantics. *unitext* columns are stored in UTF-16 encoding, regardless of the Adaptive Server default character set.

The benefits of *unitext* include:

- Large Unicode character data. Together with *unichar* and *univarchar* datatypes, Adaptive Server provides complete Unicode datatype support, which is best for incremental multilingual applications.
- *unitext* stores data in UTF-16, which is the native encoding datatype for Windows and Java environments.

  See the *System Administration Guide*.

big int Datatype Support

Adaptive Server version 15.0 supports the exact numeric datatype *bigint*.

This is the range of numbers allowed by the *bigint* datatype:

Datatype	Range of signed datatypes
bigint	Whole numbers between -2^{63} and $2^{63} - 1$ (from -9,223,372,036,854,775,808 to +9,223,372,036,854,775,807, inclusive.

Adaptive Server *bigint* support also adds the **hextobigint**, **biginttohex**, and **count_big** functions.

See the *Reference Manual: Blocks*.

Unsigned Integer Datatype Support

Adaptive Server 15.0 supports unsigned integer datatypes.

These unsigned integer datatypes allow you to extend the range of the positive numbers for the existing integer types without increasing the required storage size. That is, the signed versions of these datatypes extend both in the negative direction and the positive direction (for example, from -32 to +32). However, the unsigned versions extend only in the positive direction. This is the range for signed and unsigned datatypes:

Datatype	Range of signed datatypes	Range of unsigned datatypes
bigint	Whole numbers between -2^{63} and $2^{63} - 1$ (from -9,223,372,036,854,775,808 to +9,223,372,036,854,775,807, inclusive	Whole numbers between 0 and 18,446,744,073,709,551,615
int	Whole numbers between -2^{31} and $2^{31} - 1$ (-2,147,483,648 and 2,147,483,647), inclusive	Whole numbers between 0 and 4,294,967,295
smallint	Whole numbers between -2^{15} and $2^{15} -1$ (-32,768 and 32,767), inclusive	Whole numbers between 0 and 65535

Integer Identity

Adaptive Server 15.0 allows you to use certain datatypes as identity values.

Use these datatypes as identity values:

- *bigint*
- *int*
- *numeric*
- *smallint*
- *tinyint*
- *unsigned bigint*
- *unsigned int*
- *unsigned smallint*

Enhancements to XML Services

XML enhancements in 15.0 include XML schema support, **for xml** clause enhancements, and Unicode (I18N) support.

XML Schema Support

You can validate XML documents against either a DTD or an XML schema. The DTD or schema can be specified either in the **xmlvalidate** command or in the document itself.

You can parse, store, and query XML documents with XML schema declarations.

***for xml** Enhancements*

In Transact-SQL, an expression subquery is a parenthesized subquery. It has a single column, the value of which is the expression subquery result, and must return a single row. You can use an expression subquery almost anywhere you can use an expression.

For more information about subqueries, see the *Transact-SQL® User's Guide*.The **for xml** subqueries feature allows you to use any subquery containing a **for xml** clause as an expression subquery. For the syntax of **for xml** subqueries, see the *XML Services Guide*.

unicode Internationalization (I18N) Support

The I18N extensions fall into three categories:

- I18N support in the **for xml** clause. The columns of the result set you map to XML can contain non-ASCII data.Such data can be represented in the generated SQLX XML document either as plain characters or as numeric character representations (NCRs).
- I18N in **xmlparse** and **xmlvalidate**, to store and validate documents containing non-ASCII data.
- I18N in **xmlextract** and **xmltest**, to process XML documents and queries containing non-ASCII data.

Adaptive Server Plug-in Enhancements

Adaptive Server 15.0 includes enhancements to the Adaptive Server Plug-in.

Enhancements to the Adaptive Server Plug-in improve efficiency and convenience:

- An enterprise view that includes Server Discovery (which enables you to find available servers on the system) and automatic server status.
- The ability to update servers, administrate remote servers, and manage server logs.
- SQL Preview and Job Scheduler integration.
- A graphical query plan viewer.
- The ability to integrate external tools.

Interactive SQL

Interactive SQL allows you to execute SQL statements, build scripts, and display database data to the server.

You can run Interactive SQL individually or from the Adaptive Server Plug-in. It has been integrated in the Adaptive Server Plug-in as the standard query tool. You can use Interactive SQL to:

- Browse the information in a database.
- Test SQL statements that you plan to include in an application.
- Load data into a database and carrying out administrative tasks.

In addition, Interactive SQL can run command files or script files. For example, you can build repeatable scripts to run against a database and then use Interactive SQL to execute these scripts as batches.

User-Defined Web Services

Web Services lets you create Web services and execute SQL in Adaptive Server.

In addition to the Web methods provided by the Adaptive Server Web Services Engine, Web Services lets you create Web services and execute SQL commands in Adaptive Server Enterprise using either a Web browser or a SOAP client. These user-defined Web services use existing security and auditing control inherent in Adaptive Server Enterprise.

You can create a user-defined Web service with the **create service** command, which enables you to specify the SQL to be executed, create a first-class object for which permissions can be controlled with the **grant** command, and control whether the service can be invoked with a Web browser or a SOAP client. The ASE Web Services Engine automatically generates WSDL for user-defined Web services.

See the Adaptive Server Enterprise Web Services *User's Guide*.

Very Large Storage Support

Adaptive Server 15.0 extends the allowable number of disk devices and the allowable number of 2K blocks for each device.

In pre-15.0 versions of Adaptive Server, a virtual page is described internally in a 32-bit integer: the first byte holds the device number (**vdevno**) and the succeeding three bytes describe the page offset within the device in units of 2K bytes (the virtual page number). This architecture limits the number of devices to 256 and the size of each device to 32 gigabytes —for a maximum storage limit of 8 terabytes in the entire server.

With Adaptive Server 15.0, the device number and the page offset are stored in separate 32-bit integers. The new architecture allows you to create up to 2,147,483,647 disk devices, each of which can be as large as 2,147,483,648 2K blocks or 4 terabytes.

Note: Because of schema changes to the *sysdevices* and *sysusages* system tables, you may need to modify scripts and stored procedures that access these tables. The device identifier must now be obtained from the *vdevno* columns of *sysdevices* and *sysusages*. The *high*, *low*, and *vstart* columns of these tables no longer store the device and virtual page number—they store only the virtual page numbers.

Automatic Running of update statistics

Run the **update statistics** command automatically at times that suit your site.

Instead of manually running **update statistics** at a certain time, you can set **update statistics** to run automatically at the time that best suits your site, and avoid running it at times that hamper your system. The best time to run **update statistics** is based on the feedback from the **datachange** function. **datachange** also helps to ensure that you do not unnecessarily run **update statistics**. You can use these templates to determine the objects, schedules, priority, and **datachange** thresholds that trigger **update statistics**, which ensures that critical resources are used only when the query processor generates more efficient plans.

Because it is a resource intensive task, the decision to run **update statistics** should be based on a specific set of criteria. Some of the key parameters that can help you determine a good time to run **update statistics** are:

- How much has the data characteristics changed since you last ran update statistics? This is known as the “datachange” parameter.
- Are there sufficient resources available to run **update statistics**? These include resources such as the number of idle cpu cycles and making sure that critical online activity does not occur during **update statistics**.

Datachange is a key metric that helps you measure the amount of altered data since you last ran **update statistics**, and is tracked by the **datachange** function. Using this metric and the criteria for resource availability, you can automate the process of running **update statistics**. The Job Scheduler provides the mechanism to automatically run update statistics. Job Scheduler includes a set of customizable templates that determine when **update statistics** should be run. These inputs include all parameters to **update statistics**, the **datachange** threshold values, and the time when to run **update statistics**.

SySAM License Management

The Sybase Software Asset Management (SySAM) implementation has changed.

The changes include:

- Asset management and reporting tools are provided with SySAM version 2.0. These tools allow you to monitor license usage and compliance.
- A single installation method supports all Adaptive Server editions.
- SySAM configuration is no longer optional.
- Flexible SySAM configuration options are provided.
- SySAM licenses are no longer shipped along with order fulfillment. You must obtain license certificates from the Sybase Product Download Center (SPDC).
- SySAM license keys include information about the support plan you purchased. You must update these licenses whenever you renew your support plan.
- Licensing policies are strictly and consistently enforced.
- Adaptive Server can function under grace periods if it is not able to obtain a license. These grace periods allow customers reasonable time to respond to the issues causing license failure. Adaptive Server continues to function normally during the grace period. Adaptive Server features or the server itself will shut down at the end of the grace period if the licensing issues are not resolved.
- You can receive real-time e-mail notifications about licensing events.
- Licenses issued from SPDC include information about the host machine where the licenses will be deployed. These licenses cannot be used on another machine without being reissued from SPDC.

These changes affect the Adaptive Server installation and configuration process. See the SySAM Configuration chapter of the *Configuration Guide* for details on SySAM configuration and deployment options. See the *Adaptive Server Installation Guide* for your platform on pre-installation planning and SySAM installation information.

Plan your SySAM deployment before installing Adaptive Server.

Warning! SySAM provides for grace periods when it encounters licensing problems. When Adaptive Server enters such a grace period, the Adaptive Server error log is updated with this information. Optionally, e-mail notifications can be configured for such events. You must fix the problems causing Adaptive Server to go into grace. While Adaptive Server functions normally during this grace period, it may shutdown or disable the licensed features if the problem causing license failure is not fixed within the grace period.

Query Processing Metrics (qp Metrics)

Query processing (QP) metrics identify and compare empirical metric values in query execution. When a query is executed, it is associated with a set of defined metrics that are the basis for comparison in QP metrics.

The metrics captured include:

- CPU execution time – the time, in milliseconds, it takes to execute the query.

- Elapsed time – the difference in milliseconds between the time the command started and the current time, as taken from the operating system clock.
- Logical IO (LIO) reads – the number od Logical IO reads.
- Physical IO (PIO) reads – the number of Physical IO reads.
- Count – the number of times a query is executed.
- Abort count – the number of times a query is aborted by the resource governor due to a resource limit being exceeded.

Each metric has three values: minimum, maximum, and average. Count and abort count are not included.

Updates to Abstract Plans

Adaptive Server 15.0 supports enhancements to the query processor.

For a description of the new and changed abstract plans, see the *Query Processing Guide*.

showplan Changes

The Adaptive Server 15.0 version of **showplan** better represents the steps performed by the query processor.

Adaptive Server changes the format of the **showplan** messages to better convey the shape of the query plan. Instead of the **showplan** messages displayed in a vertical format:

```
delete
from authors
where au_lname = "Willis"
and au_fname = "Max"
```

```
QUERY PLAN FOR STATEMENT 1 (at line 1).

The type of query is DELETE.
        The update mode is direct.

        FROM TABLE
           authors
        Nested iteration.
        Using Clustered Index.
        Index : au_names_ix
        Forward scan.
        Positioning by key.
        Keys are:
           au_lname  ASC
           au_fname  ASC
        Using I/O Size 2 Kbytes for index leaf pages.
        With LRU Buffer Replacement Strategy for index leaf pages.
        Using I/O Size 2 Kbytes for data pages.
        With LRU Buffer Replacement Strategy for data pages.
```

```
        TO TABLE
           authors
```

The Adaptive Server 15.0 version of **showplan** displays a series of "pipes" (the "|" symbol) to distinguish each of the steps performed by the operators.

In the following query, there are three operators, EMIT, DELETE, and SCAN, so this query includes three sets of pipes to display this organization:

```
delete
from authors
where au_lname = "Willis"
and au_fname = "Max"
```

```
QUERY PLAN FOR STATEMENT 1 (at line 1).
2 operator(s) under root

The type of query is DELETE.
ROOT:EMIT Operator
  |DELETE Operator
  |  The update mode is direct.
  |
  |  |  |SCAN Operator
  |  |  |  FROM TABLE
  |  |  |  authors
  |  |  |  Index : aumind
  |  |  |  Forward Scan.
  |  |  |  Positioning by key.
  |  |  |  Keys are:
  |  |  |       au_lname ASC
  |  |  |       au_fname ASC
  |  |  |  Using I/O Size 8 Kbytes for index leaf pages.
  |  |  |  Using LRU Buffer Replacement Strategy for index leaf pages
  |  |  |  Using I/O Size 8 Kbytes for data pages.
  |  |  |  With LRU Buffer Replacement Strategy for data pages.
  |
  |  TO TABLE
  |   authors
  |   Using I/O Size 8 Kbytes for data pages.
```

Note: This version of Adaptive Server also includes the ability to display **showplan** messages in XML.

Secure Socket Layer Uses FIPS 140-2

In Adaptive Server 15.0, the SSL uses cryptographic modules validated for FIPS 140-2, level 1.

Secure Socket Layer (SSL) is the standard for securing the transmission of sensitive information – such as credit card numbers, stock trades, and banking transactions – over the Internet. SSL relies on public key and secret key cryptography.

The SSL used in Adaptive Server release 15.0 uses cryptographic modules validated for FIPS 140-2, level 1. The cryptographic modules are Certicom Security Builder GSE for Adaptive Server products running on Windows, Solaris, AIX and HPUX operating systems. For more information, see validation certificate #542, dated June 2, 2005 at NIST website, http://csrc.nist.gov/cryptval/140-1/1401val.htm.

System Changes for Adaptive Server Version 15.0

Adaptive Server 15.0 introduces changes in utilities, commands, system tables, functions, stored procedures, reserved words, and monitoring tables.

Utilities

Adaptive Server 15.0 supports changes to the **bcp**, **dataserver**, **sqlsrvr**, **ddlgen**, and **preupgrade** utilities.

Table 54. Changes to utility programs

Utility	Change
bcp	Adds new parameters **--sho-fi** and **--hide-vcc**, to support computed columns and functional indexes. Adds new parmeter **--maxconn** to support for parallel loading into partitioned tables. **bcp** interface has changed to now allow you to run **bcp in** and **bcp out** to and from specific partitions.
dataserver	Specifies the **-b master_database_size** parameter in terabytes.
sqlsrvr	Specifies the **-b master_database_size** parameter in terabytes.
ddlgen	Adds the WS object type for the **-T object_type** parameter to support user-defined Web services.
preupgrade	**preupgrade** includes options to perform incremental checks for various upgrade checks and is enhanced to run on a single database that is undergoing an upgrade using **load database**

See the *Utility Guide*.

Reserved Words

Adaptive Server 15.0 supports new reserved words in support of scrollable cursors and XML services.

- **insensitive** – supports scrollable cursors

- **xmlextract** – supports XML services
- **xmlparse** – supports XML services
- **xmltest** – supports XML services

You must change all database names that are new reserved words before you can upgrade from an earlier release of the server. You can change table, view, and column names or use delimited identifiers. Once you upgrade to version 15.0, you cannot use database objects whose names are new reserved words until you modify your procedures, SQL scripts, and applications.

See the Reference Manual: Blocks.

Global Variables

Adaptive Server 15.0 introduces new global variables.

Table 55. New global variables

Variable	What it displays	Value
@@rowcount	Enhanced to display the number of rows moved by a cursor, scrollable or nonscrollable. In a nonscrollable cursor, the rows are fetched from the underlying tables to the client. In a scrollable cursor, the rows counted are fetched from the current result set, not from the underlying tables.	The value of the global variable **@@rowcount** is affected by the specified cursor type. The default, non-scrollable cursor moves forward one row at a time; the maximum value is the number of rows in the result set. In a scrollable cursor the value of continues to increment, whatever the direction of the **fetch** command; there is no maximum value.
@@fetch_status	The status of a **fetch** command used for a scrollable cursor.	0 – **fetch** statement successfully executed. -1 – either the **fetch** statement failed, or the row requested is outside the result set. -2 – value reserved.

Variable	What it displays	Value
@@cursor_rows	The total number of rows in the cursor result set.	0 – No cursors are open, or no rows qualify for the last open cursor. -1 – Semi-sensitive and scrollable, but the scrolling worktable is not yet populated. The number of rows that qualify the cursor is unknown. *n* – The last opened or fetched cursor result set is fully populated; the value returned is the total number of rows in the result set.

Configuration Parameters

Adaptive Server 15.0 introduces new and changed configuration parameters.

Table 56. New configuration parameters

Function	Description
enable metrics capture	Enables Adaptive Server to capture metrics at the server level.
enable semantic partitioning	Enables semantic (hash-, list-, range-) partitioning of tables and indexes at a licensed site.
enable web services	Enables web services
enable xml	Enables the XML services
max native threads per engine	Defines the maximum number of native threads the server spawns per engine
max partition degree	Configures the amount of dynamic repartitioning Adaptive Server requires, which enables Adaptive Server to use horizontal parallelism
max resource granularity	Sets the maximum percentage of the system's resources a query can use
number of devices	Specifies the number of database devices Adaptive Server can use
number of dump threads	Controls the number of threads that Adaptive Server spawns to perform a memory dump
number of open partitions	Specifies the number of partitions that Adaptive Server can access at one time.
optimization goal	Allows you to configure for three optimization goals, which you can specify at three tiers: server level, session level, and query level

Function	Description
optimization timeout limit	Specifies the amount of time Adaptive Server can spend optimizing a query as a percentage of the total time spent processing the query
rtm thread idle wait period	Defines the time a native thread used by Adaptive Server waits when it has no work to do
sysstatistics flush interval	Determines the length of the interval (in minutes) between flushes of *sysstatistics*
statement cache size	Increases the server allocation of procedure cache memory and limits the amount of memory from the procedure cache pool used for cached statements. The statement cache feature is enabled server-wide.

Table 57. Changed configuration parameters

Parameter	Change
default network packet size	Previous versions of Adaptive Server used a **default network packet size** of 512. As of Adaptive Server version 15.0, the **default network packet size** is 2048.

See the *Administration Guide: Volume 1*.

Functions

New and changed functions for Adaptive Server 15.0

Table 58. New functions

Function	Description
biginttohex	Returns the platform-independent hexadecimal equivalent of the specified integer
count_big	Returns the number of (distinct) non-null values or the number of selected rows as a *bigint*
datachange	Measures the amount of change in the data distribution since update statistics
data_pages	Returns the number of pages used by the specified table, index, or a specific
hextobigint	Returns the *bigint* value equivalent of a hexadecimal string
is_quiesced	Returns 1 if the database is quiesced and 0 if it is not.
partition_id	Returns the partition id of the specified data or index partition name.

Function	Description
partition_name	The explicit name of a new partition, partition_name returns the partition name
reserved_pages	Reports the number of pages reserved to a table, index or a specific partition.
row_count	Returns an estimate of the number of rows in the specified table.
showplan_in_xml	Returns the execution plan in XML.
ssel_message	Returns the message text when you specify a message ID.
tran_dumpable_status	Returns a true/false indication of whether **dump transaction** is allowed.
used_pages	Reports the number of pages used by a table, an index, or a specific partition.
xmlvalidate	Validates XML documents, including those containing non-ASCII characters (I18N). Described in *XML Services*.

Several function names have been replaced with more readable names.

Table 59. Superceded functions with their new function names

Superceded function name	New function name
data_pgs	**data_pages**
used_pgs	**used_pages**
reserved_pgs	**reserved_pages**
rowcnt	**row_count**
ptn_data_pgs	**data_pages**

Commands

New and changed commands in Adaptive Server 15.0

Table 60. New commands

Command	Function
create service	For creating a user-defined Web service.
drop service	For creating a user-defined Web service.
update table statistics	Updatex *systabstats* statistics for a table or a partition.

Table 61. New set command options

set Option	Description
set delayed_commit	Allows you to determine when log records are written to disk. With the **delayed_commit** parameter set to true, the log records are asynchronously written to the disk and control is returned to the client without waiting for the IO to complete
set plan optgoal	Sets the optimization goals at the session level.
set plan opttimeoutlimit	Sets the limit the time taken by long-running and complex queries at the session level.
set metrics_capture on/off	Activates QP metrics at the session level.

Table 62. Changed commands

Command	Change
alter table	Syntax added to support computed and materialized or non-materialized columns. Adds support for paritions.
create index	Enhanced to allow computed columns to be used as index keys, in the same way as regular columns, and to create function-based indexes. Adds support for paritions.
create table	Syntax added to support computed and materialized or non-materialized columns. Adds support for paritions.
dbcc	Adds support for partitions.
declare cursor	Syntax added for scrollable cursors. Syntax added to support **semi_sensitive**, **insensitive**, and **scrollable** cursors.
delete statistics	Adds support for partitions.
disk init	The **size** parameter can be specified in terabytes. Adds the **directio** parameter, which allows you to configure Adaptive Server to transfer data directly to disk, bypassing the operating system buffer cache
disk reinit	Adds the **directio** parameter, which allows you to configure Adaptive Server to transfer data directly to disk, bypassing the operating system buffer cache
fetch	**fetch_orientation** options added to support scrollable cursors: **next**, **prior**, **first**, **last**, **absolute**, and **relative**.

Command	Change
reorg	Adds support for partitions.
select	**for xml** clauses added to support XML services. Adds support for partitions.
truncate table	Adds support for partitions.
update all statistics	Adds support for partitions.
update statistics	Adds support for partitions.
update partition statistics	Made obsolete.

System Stored Procedures

New and changed system stored procedures in Adaptive Server 15.0

Table 63. New stored procedures

Stored procedure	Function
sp_helpcomputedcolumn	Reports information on all the computed columns in a specified table
sp_version	Returns the version information of the installation scripts (`installmaster`, `installdbccdb`, and so on) that was last run and whether it was successful.

Table 64. Changed system stored procedures

Stored procedure	Change
sp_checksource	Checks the existence of computed columns source text.
sp_help	Reports information on computed columns, function-based indexes, and partitions.
sp_helpartion	Adds detailed partition information to its output.
sp_helpindex	Reports information on computed column indexes, function-based indexes, and partitions.
sp_helptext	Displays the source text of computed columns, function-based index definitions, and partitions.
sp_hidetext	Hides the text of computed columns, function-based index keys, and partition condition.

Stored procedure	Change
sp_modifylogin	Adds option **"enable logins during recovery"**.
sp_webservices	Adds **addalias**, **deploy**, **dropalias**, **listudws**, **listalias**, and **undeploy** options to support user-defined Web Services.
sp_monitorconfig	Supports the **number of open partitions** configuration parameter.
sp_countmetadata	Supports the **number of open partitions** configuration parameter.
sp_helpsegment	Prints segment bindings for objects and partitions.
sp_objectsegment	Displays segment information for all partitions for an object.
sp_placeobject	Enables future allocations for a partition from a new segment.
sp_dbcc_faultreport	Creates reports for a specific OPID or fault type.
sp_sysmon	Reports information related to open partitions Metadata Cache Management section of the configuration file.

See the *Reference Manual: Procedures*.

System Tables

New and changed system tables in Adaptive Server 15.0

Adaptive Server version 15.0 provides the necessary row-locked catalog infrastructure to support enhanced, multi-user-concurrent data-definition language (DDL) operations. However, this release does not change the catalog locking behavior for DDL operations. Applications that perform heavy multi-user DDL operations (for example, creating or dropping tables in *tempdb*, will not see any change in behaviour in this release for catalog blocking, or any increased DDL concurency.

Table 65. New system tables

Table	Description
syspartitions	*syspartitions* is completely changed from the pre-15.0 version of the table. All columns are new. *syspartitions* supports both semantic and round-robin partitioning of tables and indexes.
syspartitionkeys	Contains a row for each column in a partition key for each hash-, range-, and list-partitioned table.

Table 66. Changed system tables

Table	Change
syscolumns	New fields: • **computedcol** • **status3** New columns: • *encrtype* –Type of encryption • *encrlen* – Length of encrypted column • *encrkeydid* – Encryption key id • *encrkeydb* – Database name containing encryption key • *encrdate* – Date column was encrypted. New bits in *status2* field: • Hex: 0x00000010, Decimal 16 – the column is a computed column. • Hex: 0x00000020, Decimal 32 – the column is a materialized computed column. • Hex: 0x00000040, Decimal 64 – the column is a computed column in a view.
sysconstraints	New internal bit in status field: Hex 0x0100, decimal 265 – indicates a computed column object.
sysdevices	New columns: • *vdevno* – device identification number • *crdate* – date device created • *resizedate* – date size of device changed • *status2* – Additional status2 bits.
sysusages	New column: *vdevno* – device identification number
sysstatistics	New columns: • *indid* – index ID of the data partition. Always 0. • *partitionid* – ID of the data partition • *ststatus* – Internal status bits Unique placement index on *id*, *indid*, *partitionid*, *statid*, *colidarry*, *formatid*, *sequence*

Table	Change
systabstats	New columns: • *partitionid* – ID of data or index partition • *statmoddate* – Date when statistics were last modified on disk. • *unusedpgcnt* – Number of unused pages. • *oampagecnt* – Number of OAM pages for each partition.
syspartitions	*syspartitions* is completely changed from the pre-15.0 version of the table. All columns are new. *syspartitions* supports both semantic and round-robin partitioning of tables and indexes.
syscomments	New column: *partitionid* – ID of data or index partition Table enhanced to store the text of computed column or function-based index key expression.
sysindexes	New columns: • *partitiontype* – partitioning strategy: 1 – range, 2 – hash, 3 – round-robin, 4 – list • *conditionid* – ID of the partition condition New rows: contains one row for each function-based index or index created on a computed column. One new internal status bit added to the status2 field: Hex 0x8000, decimal 32768 – the index is a function-based index.
syslocks	New columns: • *nodeid* – Reserved for future use. • *partitionid* – ID of data or index partition. Reserved for future use. Always 0.

Table	Change
sysobjects	New object in *type* column: *N* – partition condition New column: • *identburnmax* – For an identity column, maximum burned identity value • *spacestates* – Number of space states being tracked. (Only applies for DOL tables.) • *erlchgts* – Timestamp when expected row length was last changed. (Only applies to DOL tables.) New row: one row for each computed column and function-based index key object • *type* field: type “C” added to the type field, when the object is a computed column • *status2* field: new bit added to indicate that the table contains one or more function-based indexes.
sysprocedures	Stores a sequence tree for each computed column or function-based index definition, in binary form

Table 67. Datatype changes in system table columns

System table	Changed column	Datatype changes	Identifier name
sysattributes	*object_cinfo* *char_info*	*varchar(30) null* to *varchar(255) null* *varchar(255)* to *varchar(768)*	Identifier for the object
sysaudits01 – sysaudits08	*objname*	*varchar(30) not null* to *varchar(255) not null*	Object name
syscolumns	*name*	*varchar(30) not null* to *varchar(255) not null*	Column name
	remote_name	*varchar(30) null* to *varchar(255) null*	Maps local names to remote names
sysconfigures	*name*	*varchar(80) null* to *varchar(255) null*	
sysindexes	*name*	*varchar(30) null* to *varchar(255) null*	Index for the table name
sysjars	*jname*	*varchar(30) null* to *varchar(255) null*	JAR name

System table	Changed column	Datatype changes	Identifier name
sysobjects	*name*	*varchar(30) not null* to *varchar(255) not null*	Object name
sysprocesses	*hostname*	*char(10) not null* to *varchar(30) null*	Host computer name
	program_name	*char(16) not null* to *varchar(30) null*	Name of *front_end* module
	hostprocess	*char(8) not null* to *varchar(30) null*	Host process ID number
	cmd	*char (16) not null* to *varchar(30) null*	Command or process currently being executed. Evaluation of a conditional statement, such as an **if** or **while** loop, returns.
systimeranges	*name*	*varchar(30) not null* to *varchar(255) not null*	Unique name of the time range
systypes	*name*	*varchar(30)* to *varchar(255)*	Datatype name
sysdatabases	*def_remote_loc*	*varchar(255) null* to *varchar(349) null*.	

See the *Reference Manual: Tables*.

Monitoring Tables

New and enhanced monitoring tables in Adaptive Server 15.0

Table 68. New monitoring tables

Monitoring table	Description
monOpenPartitionAcitivity	Provides monitoring information for partitions

Table 69. Changed monitoring tables

Monitoring table	Changes
monEngine	New columns for housekeeper GC task
monCachedObject	New columns for partitions
monProcessObject	New columns for partitions

See the *Reference Manual: Tables*.

Obtaining Help and Additional Information

Use the Product Documentation site and online help to learn more about this product release.

- Product Documentation at *http://sybooks.sybase.com/* – online documentation that you can access using a standard Web browser. You can browse documents online, or download them as PDFs. The Web site also has links to other resources, such as white papers, community forums, maintenance releases, and support content.
- Online help in the product, if available.

To read or print PDF documents, you need Adobe Acrobat Reader, which is available as a free download from the *Adobe* Web site.

Note: A more recent release bulletin, with critical product or document information added after the product release, may be available from the Product Documentation Web site.

Technical Support

Get support for SAP® products.

If your organization has purchased a support contract for this product, then one or more of your colleagues is designated as an authorized support contact. If you have any questions, or if you need assistance during the installation process, ask a designated person to contact Technical Support as specified in your contract:

- Sybase Technical Support or the Sybase subsidiary in your area
- SAP Technical Support

Customers with an SAP support contract can obtain support for this product at the SAP support site, *http://service.sap.com/sybase/support*. You can also find information about Sybase support migration at this location (login may be required).

Customers with a Sybase support contract can obtain support for this product at *http://www.sybase.com/support* (login required).

Downloading Product Updates

Get maintenance releases, support packages and patches, and related information.

- If you purchased the product directly from Sybase or from an authorized Sybase reseller:
  a) Go to *http://www.sybase.com/support*.
  b) Select **Support > EBFs/Maintenance**.

c) If prompted, enter your MySybase user name and password.
d) (Optional) Select a filter, a time frame, or both, and click **Go**.
e) Select a product.

   Padlock icons indicate that you do not have download authorization for certain EBF/Maintenance releases because you are not registered as an authorized support contact. If you have not registered, but have valid information provided by your Sybase representative or through your support contract, click **My Account** to add the "Technical Support Contact" role to your MySybase profile.
f) Click the **Info** icon to display the EBF/Maintenance report, or click the product description to download the software.

- If you ordered your product under an SAP contract:
  a) Go to *http://service.sap.com/swdc* and log in if prompted.
  b) Select **Search for Software Downloads** and enter the name of your product. Click **Search**.

Product and Component Certifications

Certification reports verify Sybase product performance on a particular platform.

To find the latest information about certifications:

- For partner product certifications, go to *http://www.sybase.com/detail_list?id=9784*
- For platform certifications, go to *http://certification.sybase.com/ucr/search.do*

Accessibility Features

Accessibility ensures access to electronic information for all users, including those with disabilities.

Documentation for this product is available in an HTML version that is designed for accessibility.

Vision impaired users can navigate through the online document with an adaptive technology such as a screen reader, or view it with a screen enlarger.

HTML documentation has been tested for compliance with accessibility requirements of Section 508 of the U.S Rehabilitation Act. Documents that comply with Section 508 generally also meet non-U.S. accessibility guidelines, such as the World Wide Web Consortium (W3C) guidelines for Web sites.

Note: You may need to configure your accessibility tool for optimal use. Some screen readers pronounce text based on its case; for example, they pronounce ALL UPPERCASE TEXT as initials, and MixedCase Text as words. You might find it helpful to configure your tool to announce syntax conventions. Consult the documentation for your tool.

For information about how Sybase supports accessibility, see the Sybase Accessibility site: *http://www.sybase.com/products/accessibility*. The site includes links to information about Section 508 and W3C standards.

You may find additional information about accessibility features in the product documentation.

Index

? (question marks) dynamic parameter 48
@@lastlogindate global variable 133
@@lwpid global variable 74
@@lwpid, global variable 48
@@plwpid global variable 74
@@plwpid, global variable 48

A

abstract plans 136
    enhancements 135
    query-level settings 135
    saving 43
    updates to 153
Adaptive Server Plug-in 149
    Interactive SQL 149
    updates 101
aggregate processing 111
allow updates to system tables configuration parameter 32
alter {precomputed result set | materialized view } command 28
alter database 56
    async_init parameter 27
    noasync_init command 28
alter encryption key command 56
alter login command 56
alter login profile command 56
alter object modify owner command 42
alter table 28, 44
    log off parameter 44
    no datacopy parameter 26
    shrinking log 44
    split, merge, or move partitions 22
alter...modify owner command 56
Application Functionality Configuration Group 39
archive database 77
archive database access 108
asterisk (*) in nested select statements 51
asynchronously initializing databases 27
auditing
    changes 95
    failed logins 118
    hidden passwords 117
authmec() function 132
automatic master key access configuration parameter 63

B

backup enhancements 3
Backup Server 84
    binary 94
    with dump configuration command 25
backups
    database 84
    transaction 84
bcp utility 34, 155
    fast-logged 24
big int datatype 147
bigdatetime datatype 87
biginttohex function 158
bigtime datatype 87
builtin date strings configuration parameter 93
bulk copy, fast-logged bcp 24

C

cache_usage function 88
chained transactions
    changes to commands and system procedures 51
character
    field, padding with str 46
    sets, Simplified Chinese and Japanese 110
charindex function 59
charlength function 59
cis idle connection timeout configuration parameter 138
cluster configuration file, backing up 25
Cluster Edition
    archive database 77
    feature and platform matrix 75
column default cache size configuration parameter 63
columns
    creating nonmaterialized, non-NULL 47
    data-only locked (DOL) 52
    names and quoted identifiers 53
    row offset 52

commands
alter {precomputed result set | materialized view } 28
alter database 56
alter database ... noasync_init 28
alter encryption key 56
alter login 56
alter login profile 56
alter table 28, 49
alter...modify owner 56
changes for chained transactions 51
create {precomputed result set | materialized view } 28
create database 56
create database ... noasync_init 28
create encryption key 56
create login 56
create login profile 56
create table 28
create table ... [in row [(length)] | off row] 56
deallocate locator 56
declare cursor ... [release_locks_on_close]] 56
declare cursor ... release_locks_on_close 50
drop {precomputed result set | materialized view } 28
drop encryption key 56
drop login 56
drop login profile 56
dump configuration 28
dump database 28
dump database ... with shrink_log 56
dump transaction 49
grant 21, 28
grant role 28
like 56
load database 28
load transaction 28
load transaction ... listonly=create_sql 28
merge 28, 49, 56
refresh {precomputed result set | materialized view } 28
reorg rebuild 49
reorg rebuild ... with online 28
revoke 21, 28
select for update 56
select into 49
select into ... [in row [(length)] | off row] 56
set 28, 56
truncate 28
truncate lob 56
update statistics ... [, [no | partial |] hashing 28
where 56
component integration system (CIS)
end-to-end Kerberos authentication 41
compressed, shared memory dumps
and sp_shmdumpconfig 28
compressing data 40
compression
in-row large object 27
levels 84
computed columns 145
difference between materialized and not materialized 146
materialized 146
materialized and nonmaterialized 145
not materialized 146
vs function-based indexes 146
concatenation operators 56
concurrent access with reorg rebuild ... online 22
concurrent dump transaction and dump database 24
configuration file, backup up 25
configuration parameters
allow updates to system tables 32
automatic master key access 63
column default cache size 63
disable varbinary truncation 49, 63
dump history filename 25, 32
enable async database init 32
enable concurrent dump tran 32
enable dump history 25, 32
enable functionality group 39, 50, 53, 54, 63
enable hp posix async i/o 63
enable inline default sharing 39
enable permissive unicode 39
enable plan sharing 32
enable predicated privileges 32
enforce dump configuration 25, 32
kernel mode 63
kernel resource memory 63
lock timeout pipe active 48, 63
lock timeout pipe max messages 48, 63
memory dump compression level 28, 32
number of disk tasks 63
number of network tasks 63

quoted identifier enhancements 39
select for update 39
SQL Perfmon Integration 32
streamlined dynamic SQL 39, 54
syb_sendmsg port number 32
unicode noncharacters 53
update statistics hashing 32
cost of a cpu unit configuration parameter 130
cost of a logical io configuration parameter 130
cost of a physical io configuration parameter 130
count_big function 158
create {precomputed result set | materialized view } command 28
create database 44, 56
noasync_init command 28
create database configuration parameters
async_init parameter 27
enable async database init 27
create encryption key command 56
create login command 56
create login profile command 56
create service command 159
create table 28, 44
[in row [(length)] | off row] 56
command 99
deferred 22
create_locator function 59
creating deferred tables 22
cursors
locking with select for update 46
releasing cursor locks at cursor close 50
scrollable 146
curunreservedpgs function 30

D

data compression
large object (LOB) 40
regular data. 40
data load optimization 4
data_pages function 30, 158
data-only locked (DOL) columns 52
database dump 108
database size
maximum 26
datachange function 158
datalength function 59
dataserver utility 34, 155
datatypes
unitext 147
db_attr function 88
DDL, fully recoverable 49
ddlgen utility 124, 155
deallocate locator command 56
declare cursor ... [release_locks_on_close]] command 56
defaults, sharing 47
deferred compilation 109
stored procedures 115
deferred name resolution 85
configuration parameter 93
defrag 7, 19
descriptors, discarding 47
detach transaction command 100
disable varbinary truncation configuration parameter 63
disk devices, very large storage support 150
distributed transaction management (DTM) 100
DOL 52
dol_downgrade_check function 59
drop {precomputed result set | materialized view } command 28
drop encryption key command 56
drop login command 56
drop login profile command 56
drop service command 159
dropping columns
without a data copy 26
dsedit utility 98
DTM
external rollbacks 100
Transaction Manager 100
dual master system key 41
dump configuration command 28
and Backup Server 25
creating, modifying, and listing dump configurations, 25
dump configuration group 25
dump database 28
concurrent with dump transaction 24
with a dump configuration command 25
with a dump history file 25
with shrink_log 56
dump enhancements 3
dump header
dump with listonly command 25
dump history file 25
backing up 25

dump history filename configuration parameter 25
enable dump history configuration parameter 25
sp_dump_history 25
dump history filename configuration parameter 25, 32
dump transaction
recoverable operations 49
with a dump history file 25
with the dump configuration command 25
dump with listonly command
create_sql 25
load_sql 25
dumps
compressed, shared memory 28
dynamic parameters, analyzing 48
dynamic partition elimination coverage, in show_cached_plan_in_xml output 23

E

employee lifecyle management 42
enable async database init configuration parameter 27, 32
enable concurrent dump tran configuration parameter 32
enable dump history configuration parameter 25, 32
enable encrypted columns configuration parameter 130
enable functionality group configuration parameter 39, 50, 54, 63
enable hp posix async i/o configuration parameter 63
enable literal autoparm configuration parameter 138
enable merge join configuration parameter 130
enable metrics capture configuration parameter 157
enable plan sharing configuration parameter 32
enable predicated privileges configuration parameter 32
enable semantic partitioning configuration parameter 157
enable web services configuration parameter 157
enable xml configuration parameter 157
encrypted columns, enhancements to 107
end-to-end Kerberos authentication
message confidentiality 41
message integrity 41
mutual authentication 41
enforce dump configuration, configuration parameter 25, 32
exists and select statements 51
external login passwords 43

F

fast bcp, fully logged 24
feature and platform matrix 1, 75, 81
feature availability, for supported platforms 1, 75, 81
FIPS 140-2 85
and SSL 154
function-based indexes 145
vs computed columns 146
functions
charindex 59
charlength 59
create_locator 59
curunreservedpgs 30
data_pages 30
datalength 59
dol_downgrade_check 59
lct_admin 30
locator_literal 59
locator_valid 59
lprofile_id 59
lprofile_name 59
patindex 59
reserved_pages 30
return_lob 59
setdata 59
show_cached_plan_in_xml 59
show_cached_text 30
show_cached_text_long 30
show_dynamic_params_in_xml 48, 59
str 59
textptr 59
textvalid 59
used_pages 30

G

global variables 156
@@lwpid 48, 74
@@plwpid 48, 74
grant command 21, 28

grant role command 28
granular permissions 21
groups, tempdb 87

H

hash-based statistics 25
hashbytes() function 132
hashed tables 99
hextobigint function 158
hidden text 43
high availability 100
    and passwords 118
histograms
    interpolation 115
    selectivity estimates 115
    viewing with sp_showoptstats 49
housekeeper task 97
huge pages 99

I

IBM Tivoli Storage Manager 84
image datatype, storing 44
in-memory
    databases 83
    system procedures 89
    temporary databases 83
in-row
    large object compression 27
    LOB columns, storing 44
incremental data transfer 86
indent identifier, changes 53
index names
    quoted identifiers 53
index_name() function 132
initializing databases asynchronously 27
inline defaults, sharing 47
ins_by_bulk 4, 7, 19
installing system stored procedures 100
integer identity, datatypes as identity values 148
Interactive SQL 150
is_quiesced function 158
isdate function 138
isnumeric function 138
ISO 8601 duration 42

J

Java in Adaptive Server 102
joins, relocated 121

K

Kerberos
    authentication 98
    end-to-end authentication 41
    unified login authentication 41
kernel
    and CPUs 39
    process mode 39
    thread pools 39
    threaded mode 39
kernel mode configuration parameter 63
kernel resource memory configuration parameter 63
keywords 155

L

large identifiers 145
large objects (LOBs)
    data, compressing 40
    in row LOB 44
    LOBs as variables 45
    off row LOB 44
    using LOB locators in T-SQL statements 45
    where clause extension 46
latency, reducing 54
lct_admin function 30
LDAP server authentication 97
LDAPS 97
like command 56
    pattern matching 53
Linux pSeries features 99
lio and pio coverage, in show_cached_plan_in_xml output 23
literal parameterization 137
load database command 28
load enhancements 3
load transaction command 28
    listonly=create_sql 28
LOB
    compression 27
    locators, using in T-SQL statements 45
locator_literal function 59
locator_valid function 59
lock timeout pipe active configuration parameter 63

lock timeout pipe max messages configuration parameter 63
lock timeouts, monitoring 48
locks
releasing cursor locks at cursor close 50
log, shrinking 44
login triggers 116
logins
mapping 98
profiles 42
securing 42
lprofile_id function 59
lprofile_name function 59

M

master sytem key 41
matierializng deferred tables 22
max native threads per engine configuration parameter 157
max online q engines configuration parameter 130
max partition degree configuration parameter 157
max resource granularity configuration parameter 157
maximum database size 26
maximum nesting level configuration parameter 104
memory dump compression level configuration parameter 28, 32
merge command 28, 49, 56
merging partitions 22
metadata cache, discarding descriptors 47
metrics elap max configuration parameter 130
metrics exec max configuration parameter 130
metrics lio max configuration parameter 130
metrics pio max configuration parameter 130
min pages for parallel scan configuration parameter 130
mnc_full_index_filter configuration parameter 104
monCachedProcedures monitoring table 67
monCachedStatement monitoring table 35, 50, 67
monCachePool monitoring table 67
monDeadLock monitoring table 67
monDeviceSpaceUsage monitoring table 67
monErrorLog monitoring table 67
monitoring tables 78, 118
and the statement cache 119
monCachedProcedures 67
monCachedStatement 50, 67
monCachePool 67
monDeadLock 67
monDeviceSpaceUsage 67
monErrorLog 67
monLockTimeout 67
monLockTimeouts 48
monOpenObjectActivity 67
monOpenPartitionActivity 67
monProcess 67
monProcessActivity 67
monProcessLookup 67
monProcessProcedures 67
monTableColumns 67
monTables 67
monWaitClassInfo 67
monWaitEventInfo 67
monLockTimeout monitoring table 67
monLockTimeouts monitoring table 48
monOpenObjectActivity monitoring table 67
monOpenPartitionActivity monitoring table 67, 166
monProcedureCacheMemoryUsage monitoring table 142
monProcedureCacheModuleUsage. monitoring table 142
monProcess monitoring table 67
monProcessActivity monitoring table 67
monProcessLookup monitoring table 67
monProcessProcedures monitoring table 67
monSQLRepActivity monitoring table 105
monSQLRepMisses monitoring table 105
monTableColumns monitoring table 67
monTables monitoring table 67
monTableTransfer monitoring table 93
monWaitClassInfo monitoring table 67
monWaitEventInfo monitoring table 67
moving partitions 22

N

name length
remote backup server 17
nested select statements with asterisk (*) 51
no datacopy parameter 26
noncharacters, Unicode 53
nonmaterialized, non-NULL columns, creating 47
number of devices configuration parameter 157
number of disk tasks configuration parameter 63
number of dump threads configuration parameter 157

number of network tasks configuration parameter 63
number of open partitions configuration parameter 157

O

object descriptors, discarding 47
object ownership, transfering 42
object_attr function 88
off-row columns, storing 44
openssl utility 94
optdiag utility 34
optimization goal
    ins_by_bulk 4
optimization goal configuration parameter 157
optimization goals, user-defined 26
optimization timeout limit configuration parameter 157
optimizer settings, viewing 116
optimizer, deferred compilation 109

P

parallelism in Adaptive Server 144
parameters, dynamic 48
partition_id function 158
partition_name function 158
partition_object_id function 138
partitioning
    and query processor 144
    hash 143
    range 143
    round-robin 143
    support 143
partitions
    compressing 40
    merging 22
    moving 22
    spliting 22
password_random function 102
passwords 98
    management 42
    security 117
patindex function 59
pattern matching, like 53
pci memory size configuration parameter 104
permissions
    granting predicated privileges 21
    granular 21
Pluggable Authentication Module (PAM) 116
precomputed result sets 24
predicated privileges 21
preupgrade utility 155
privacy, adding predicated privileges 21
privileges
    granting 21
    predicated 21
    revoking 21
process mode 39
profiles, login 42
pssinfo function 102

Q

queries, slow-running 109
query plans 115, 144
    analyzing 55
    format 153
    shared 27
query processing
    latency, reducing 54
    metrics 152
query processor 144
question marks, dynamic parameter 48
quoted identifiers 53

R

refresh {precomputed result set | materialized view }
    command 28
reinit_descriptors subcommand 97
relaxed-durability databases 83
    system procedures 89
remote backup server name length limit 17
reorg defrag 4
reorg rebuild command
    with online parameter 22, 28
reorganization 4
Replication Server 97
reserved words 155
reserved_pages function 30, 158
restore enhancements 3
result sets, precomputed 24
return_lob function 59
revoke command 21, 28
Rivest-Shamir-Adleman (RSA) keypair
    regeneration period 42

roles, securing 42
row_count function 158
row-level locking, system tables 119
row-locked system catalogs 144
rows
    transferring 49
    variable length 52
rtm thread idle wait period configuration parameter 157

S

scalar aggregation 112
scan coverage, in show_cached_plan_in_xml output 23
scrollable cursors 146
Secure Socket Layer (SSL) 117
    and FIPS 140-2 154
security features
    dual control of encryption keys 41
    employee lifecycle management 42
    encrypted columns 107
    end-to-end Kerberos authentication 41
    external login passwords and hidden text 43
    login profiles 42
    securing logins, roles, and password management 42
    transfering object ownership 42
    unattended startup 41
select for update
    and cursors 46
    command 56
    locking rows 46
select into ... [in row [(length)] | off row] command 56
select statements, nested 51
send doneinprov tokens configuration parameter 130
set command 28, 56, 109, 123
    for fast logged bcp 24
set...opt critera command 137
setdata function 59
shared memory 99
shared query plans 27
sharing inline defaults 47
show cached plans in XML 46
show_cached_plan_in_xml
    function 46
    output 23
show_cached_plan_in_xml function 59
show_cached_text function 30
show_cached_text_long function 30
show_dynamic_params_in_xml function 48, 59
showplan changes 153
showplan_in_xml function 158
shrinking log space 44
size of statement cache 23
slow-running queries 109
sort order 110
sp_checksource system procedure 31
sp_config_dump system procedure 31
sp_dboption system procedure 31, 60
sp_defrag_status 9, 20
sp_deletesmobj system procedure 89
sp_displaylogin system procedure 60
sp_displayroles system procedure 60
sp_downgrade system procedure 125
sp_dump_history system procedure 31
sp_encryption system procedure 60
sp_help system procedure 31
sp_helpcomputedcolumn system procedure 161
sp_helpconstraint system procedure 60
sp_helprotect system procedure 31, 60
sp_helptext system procedure 31
sp_helpuser system procedure 60
sp_hidetext system procedure 31
sp_jreconfig system procedure 103
sp_listener system procedure 98
sp_locklogin system procedure 60
sp_merge_dup_inline_default system procedure 60
sp_opt_querystats system procedure 55, 60
sp_optgoal system procedure 31
sp_passwordpolicy system procedure 60
sp_querysmobj system procedure 89
sp_securityprofile system procedure 60
sp_serveroption system procedure 60
sp_shmdumpconfig
    and shared memory dumps 28
sp_shmdumpconfig system procedure 31
sp_showoptstats system procedure 60
sp_spaceusage system procedure 125
sp_tabsuspectptn system procedure 103
sp_version system procedure 161
splitting partitions 22
sproc optimize timeout limit configuration parameter 138
spt_TableTransfer system table 93
SQL
    lightweight procedurs (LWPs) 54

statement replication 97
SQL Perfmon Integration configuration parameter 32
SQL TEXT in abstract plans 43
sqlsrvr utility 155
square brackets ([]) and like pattern matching 53
ssel_message function 158
SSL
certificate, common name 98
support 117
standard deviation 111
startup delay configuration parameter 138
statement cache 46
and monitoring tables 119
maximum size 23
saving abstract plans 43
statement cache size configuration parameter 157
statements, maximum size 23
statistical aggregate functions 110
statistics
gathering 151
hash-based 25
viewing with sp_showoptstats 49
stored procedures, deferred compilation 109
str function 59
padding character field 46
streamlined dynamic sql configuration parameter 54
switches, view settings 44
syb_sendmsg port number configuration parameter 32
Sybase Central 149
sybdiag utility 54, 67
sybperf utility 67
sysaltusages system table 33
SySAM changes 151
sysattributes system table 33, 65
syscacheinfo system table 65
syscachepoolinfo system table 65
syscolumns system table 65
syscomments system table 65
sysdatabases system table 93
sysdevices system table 93
syslocks system table 33
syslogins system table 65
syslogshold system table 33
sysobjects system table 33
sysoptions system table 44, 65, 128
syspartitionkeys system table 162
syspartitions system table 33, 162
syspoolinfo system table 65
sysprotects system table 33
sysservers system table 65
syssrvroles system table 65
sysstatistics flush interval configuration parameter 157
systabstats system table 33
system encryption password, replaced 41
system keys
dual master 41
master 41
system procedures
changes for chained transactions 51
quoted identifiers 53
sp_checksource 31
sp_config_dump 31
sp_dboption 31, 49, 60
sp_displaylogin 60
sp_displayroles 60
sp_dump_history 31
sp_encryption 60
sp_help 31
sp_helpconstraint 60
sp_helprotect 31, 60
sp_helptext 31
sp_helpuser 60
sp_hidetext 31
sp_locklogin 60
sp_merge_dup_inline_default 60
sp_opt_querystats 55, 60
sp_optgoal 31
sp_passwordpolicy 60
sp_securityprofile 60
sp_serveroption 60
sp_shmdumpconfig 31
sp_showoptstats 49, 60
system tables
row-locked 144
sysaltusages 33
sysattributes 33, 65
syscacheinfo 65
syscachepoolinfo 65
syscolumns 65
syscomments 65
syslocks 33
syslogins 65
syslogshold 33
sysobjects 33

sysoptions 65
syspartitions 33
syspoolinfo 65
sysprotects 33
sysservers 65
syssrvroles 65
systabstats 33
systhresholds 33
sysusages 33
systhresholds system table 33
sysusages system table 33

T

T-SQL statements, using LOB locators 45
tables
compressing 40
deferred creation 22
deferred tables, creating 22
quoted identifiers 53
transfering rows 49
tempdb groups 87
text datatype, storing 44
textptr function 59
textvalid function 59
thread pools 39
threaded mode 39
Tivoli Storage Manager 84
system procedures 89
trace flags, viewing currently set 44
trailing zeros, truncating 49
tran_dumpable_status function 158
transactions
chained 51
locking with select for update 46
mode 51
transfer table command 91
transferring
data 86
rows 49
triggers, instead of 122
truncate {precomputed result set | materialized view } command 28
truncate lob command 56

U

Unicode
enhancements 149
noncharacters, ignoring 53
unitext
datatype, storing 44
support 147
unsigned integer datatypes 148
update statistics 151
[, [no | partial |] hashing 28
hash-based 25
update statistics hashing configuration parameter 32
update table statistics command 159
updates
locking with select for update 46
used_pages function 30, 158
user-defined
functions 121
optimization goals 26
utilities
bcp 34
dataserver 34
optdiag 34
sybdiag 54, 67
sybperf 67

V

variable-length rows, expanded 52
variables, using LOBs 45
variance 111
vector aggregation 112
very large storage support (VLSS) 150
views
quoted identifiers 53
virtually hashed tables 99

W

Web services 150
where clause, extension 46
where command 56
worktables, in show_cached_plan_in_xml 23

X

XML
enhancements 149
internationalization support 149
schema support 149
showing cached plans 46

viewing statistics and histograms 49
xmltable() function 120
xmlvalidate function 158

Z

zap defrag info 7, 19
zeros, truncating 49